DOCUMENT RESUME

ED 375 148 TM 022 094

AUTHOR Derry, Sharon; And Others
TITLE Psychological Foundations of the TiPS System: A Handbook for System 1.0.
INSTITUTION Wisconsin Center for Education Research, Madison.
SPONS AGENCY Office of Naval Research, Washington, D.C.
PUB DATE Jun 94
CONTRACT ONR-N00014-93-1-0310
NOTE 61p.; The figure and some graphics may not reproduce well. Paper presented at the Annual Meeting of the American Educational Research Association (New Orleans, LA, April 4-8, 1994).
PUB TYPE Reports - Evaluative/Feasibility (142) -- Speeches/Conference Papers (150)

EDRS PRICE MF01 Plus Postage. PC Not Available from EDRS.
DESCRIPTORS *Cognitive Psychology; *Computer Assisted Instruction; Educational Environment; Educational Philosophy; Intelligent Tutoring Systems; *Mathematics; *Problem Solving; Secondary Education; Training; Tutoring
IDENTIFIERS Cognitive Apprenticeships; *Tutoring in Problem Solving

ABSTRACT

TiPS (Tutoring in Problem Solving) System 1.0 is a computer-based instructional environment designed to support adolescent and adult problem-solving training based on the instructional objectives and training theory described in this document. The objectives of TiPS training are basic mathematics concepts and general problem-solving skills derived from cognitive theory, and the training philosophy is based on cognitive apprenticeship. The TiPS approach employs complex arithmetic story problems as contexts for helping students develop better problem-solving abilities. Future versions of TiPS will incorporate expanded intelligent diagnostic and tutoring capabilities, and thus will be able to tutor students with a minimum of human intervention. The current version works best if human instructors oversee student work on the system. Instructional personnel can use this handbook to help them understand the TiPS system, curriculum, and instructional philosophy. An appendix gives excerpts from the user's guide. (Contains 56 references, 1 table, and 1 figure.) (Author)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

U.S. DEPARTMENT OF EDUCATION
Office of Educational Research and Improvement
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)
☑ This document has been reproduced as received from the person or organization originating it
☐ Minor changes have been made to improve reproduction quality
• Points of view or opinions stated in this document do not necessarily represent official OERI position or policy

"PERMISSION TO REPRODUCE THIS MATERIAL IN OTHER THAN PAPER COPY HAS BEEN GRANTED BY
KEITH TOOKEY
TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)"

ED 375 148

June 1994

Psychological Foundations of the TiPS System: A Handbook for System 1.0

Sharon Derry and TiPS Project Staff:

Keith Tookey, Chris Smith, Michael Potts, Don Wortham, and Afroditi Michailidi

Project Supported by ONR Contract N00014-93-1-0310

Wisconsin Center for Education Research
School of Education, University of Wisconsin–Madison

BEST COPY AVAILABLE

TM022094

Table of Contents

Abstract 1

Introduction 2

Arithmetic and General Problem Solving Skills 3
Arithmetic Story Problems as Training Context 6
Context for Studying Transfer 7

The TiPS Curriculum 8

Objective 1: Schemas of Problem Solving 9
- Schema Types 10
- Position of Unknown and Schema Difficulty 10
- Schema Understanding 11
- Schemas and Complex Problems 11

Objective 2: Strategic Planning 11
- Working Backward 12
- Working Forward 14

Objectives 3, 4, & 5: Metacognition 16
Objectives 6, 7, & 8: The Problem of Inconsiderate Text 17
Objective 9: Problem Solving as Social Process 18
Objective 10: Memory Management and Multiple Representations 19

TiPS Philosophy: How Should Problem Solving Be Taught? 21

Cognitive Apprenticeship 21
Tutoring Styles 23
Cognitive Apprenticeship Through Worked Examples 26
Instructional Routines 28

Routines for Tutoring: Suggestions for Tutors 30

Schema Learning 30
- Schema Analysis Routine 30
- Problem-Writing Routine 31

Routines for Strategy Learning 31
- Strategy-Demonstration Routine 31
- Overcoming Difficulties by Working Backward 31

Routines for Error Checking, and Comprehension Monitoring 32
- Self-Monitoring Routine 32
- Monitoring of Others Routine 32

Modeling Supportive Beliefs 33
Selective Encoding Routine 33
Search and Integration Routine 34
Cognitive Negotiation Routines 34
- Demonstrating Negotiation Skills 34
- Reflecting on Negotiation Skills 34

Tutoring with Problem and Worked Examples Banks 35

Structure of the Problem Directory 35
Structure of the Worked-Examples Directory 36

Tutoring with TiPS 1.0: Concluding Comments 37
TiPS as a Generalized Design 38

References 39

Table 1
Subdirectories of the Problem Bank 35b

Figure 1
TiPS Screen 12b

Appendices

Appendix A, Excerpts from TiPS On-line User's Guide
Contents A1
Introduction A2
Schema Definitions A3
Schemas in Problems (example problem) A5
Commands for Creating Solutions A6
The Problem Window A8
Problem Files A9
Appendix B, Example TiPS Problems
Appendix C, Skills of Cooperative Problem Solving

Psychological Foundation of the TipS System:

A Handbook for System 1.0

Abstract

TiPS (Tutoring in Problem Solving) System 1.0 is a computer-based instructional environment designed to support adolescent and adult problem-solving training based on the instructional objectives and training theory described in this document. The objectives of TiPS training are basic mathematics concepts and general problem-solving skills derived from cognitive theory, and the training philosophy is based on cognitive apprenticeship. The TiPS approach employs complex arithmetic story problems as contexts for helping students develop better problem-solving abilities. Future versions of TiPS will incorporate expanded intelligent diagnostic and tutoring capabilities, and thus will be able to tutor students with a minimum of human intervention. The current version works best if human instructors oversee student work on the system. Instructional personnel can use this handbook to help them understand the TiPS system, curriculum, and instructional philosophy.

TiPS (Tutoring in Problem Solving) Version 1.0 is a computer-based instructional environment designed to support adolescent and adult problem-solving training programs based on objectives and instructional methods substantially similar to those to be described in this document. Both individualized and small-group (e.g., pairs working together on the system) training can be supported. Although the TiPS architecture can be generalized to problem-solving training in different subject domains, the current implementation focuses on complex arithmetic story problems as contexts for training relatively general problem-solving skills.

Students typically interact with TiPS 1.0 by first selecting a story problem exercise from the problem bank, which contains word problems representing a range of basic math concepts, real-world contexts, and levels of challenge, and then by analyzing and solving the selected problem using the graphics interface and other supportive tools provided by the system. A "starter" bank of about 100 complex arithmetic story problems suitable for military and civilian problem-solving training has been included in the current version of TiPS, although instructors and other authorized users can update and edit the problem bank and input new problem exercises as desired. Further information on current system features is given in the example screen and partial printout of the TiPS Help System, both provided as Appendix A. Example problems from the current TiPS problem bank are attached as Appendix B.

TiPS is under continuous development as a number of upgrades are being planned and implemented. During the current funding year (Feb 1, 1994 - January 31, 1995), the following system features will be added to those shown in Appendix A: (1) an improved help system that will be context sensitive, (2) on-line reference material to support problem solving, (3) dynamic think-aloud worked examples illustrating expert problem solving and emphasizing the skills listed in the TiPS curriculum, (4) local student modeling, allowing for context-sensitive performance feedback. During funding year 3 we will incorporate the following: (5) global student modeling permitting system-controlled curriculum-level planning and summary evaluative feedback on progress toward objectives, (6) system-controlled interactive tutoring, and (7) additional graphics interface tools prompting various levels of conceptual analysis. In addition to the system capabilities listed above, a number of interface improvements are being made.

Although future versions of TiPS will be capable of tutoring students with less human intervention, the current version works best if a human instructor oversees

student work on the system. With human tutorial support, the current system can be used in a great many creative ways to support instruction in problem solving. In this report, we overview the knowledge that should be understood by TiPS instructors, especially those in charge of supervising implementations of the TiPS 1.0 system in adult learning centers. Supervising instructors and tutors can use this report to help them

- learn about the skills curriculum supported by TiPS,
- acquire basic understanding of TiPS system capabilities,
- understand the three stages of the cognitive apprenticeship tutoring model recommended for TiPS instruction,
- employ tutoring styles appropriate for each of the three stages of cognitive apprenticeship,
- learn how to tutor students with TiPS using worked examples exercises, and
- acquire ideas for different tutoring exercises and routines.

The ideas set forth in this document correspond to ways that TiPS is being used in our research to support experimental problem-solving training in adult educational environments. Because instructional methods and supporting system features represent cognitive hypotheses regarding problem-solving training and tutorial instruction, both scientific theory and the TiPS system are being developed and tested simultaneously. Consequently, some recommended procedures may evolve and change during the course of our project. Also, as we study how expert human tutors use TiPS Version 1.0 to support cognitive-apprenticeship tutoring in actual educational environments, we will endeavor to model and incorporate key features of these interactions into system-controlled routines, thus relieving human instructors of much direct tutoring responsibility and enabling them to supervise more students.

Arithmetic and General Problem-Solving Skills

One of our major goals is to perfect TiPS training to the point where it will be powerful enough to improve real-world problem solving. This transfer problem is difficult and past attempts to train general problem-solving skills directly have not met with astounding success (see Derry, 1989, for discussion; also, Chipman, Segal, & Glaser, 1985; Segal, Chipman, & Glaser, 1985). Yet, there is evidence that abstract concepts and skills taught in certain general problem-solving domains do improve performance in other out-of-domain contexts. For example, instruction in algebra

improves physics problem solving (Bassok & Holyoak, 1989), and instruction in statistics improves performance on tests of informal reasoning (Nisbett, Long, Lehman, & Cheng, 1987). Moreover, there is experimental evidence that spontaneous transfer can be achieved through training that focuses on relevant underlying schematic structures in training problems (Catrambone & Holyoak, 1989).

Our TiPS project is based on the hypothesis that schema-based problem-solving training contextualized within the domain of arithmetic can transfer to out-of-training contexts, producing improvements in both academic and nonacademic problem solving. Our transfer hypothesis can be stated specifically as follows: *It is possible to engineer formal computer-supported tutorial training in arithmetic problem- solving concepts and skills that will significantly improve students' real-world problem-solving ability.* By *computer-supported tutorial training* we mean instruction that is largely computer supported but that may incorporate some human intervention and that can feasibly be implemented within the context of a typical classroom or learning center environment. *Real-world problem solving* refers to nontrivial problems that occur naturally in job-related or daily living contexts.

The word *formal* in our hypothesis statement refers to instruction that attempts to impart abstract concepts and thinking skills on the faith that students will later transfer and employ them in situations beyond the training environment (Elshout, 1991). A contrasting view is represented by some situated cognition positions (e.g., Allen, 1991; Lave & Wenger, 1991) -- those arguing that formalisms cannot be abstracted and transferred beyond the social and physical settings in which they evolve. Support for situated cognition is garnered from anthropological studies of real-world problem solving (e.g., Lave & Wenger, 1991; Saxe, 1988), which show that, when faced with a problematic situation, humans often do not apply concepts learned in school (such as fractions), but rather adapt objeccts from their physical environments and employ them as tools for structuring and understanding of the problem at hand. For example, Scribner (1984) observed that workers packing dairy products used the structure of their packing crates, rather than formal mathematics, to decide how to sort goods into shipping containers. In addition, studies indicate that problem solving in work environments takes place through special forms of social interaction, *cognitive apprenticeships,* whereby mentors assist and guide the problem-solving activity of their less advanced colleagues. Although we dispute neither these findings nor the concept of situated cognition, we reject the notion, sometimes associated with situated cognition, that learning and problem solving in the real world are so profoundly different from classroom practice that little transfer is possible.

Rather, we agree with Merrill (1992), who argues that the direct training of abstractions is both efficient and possible, capitalizing on an important ability that discriminates human from other forms of animal intelligence: the ability to "mindfully abstract" (Salomon & Perkins, 1989) concepts and procedures from one setting and use them as cognitive tools in another. But if this is true, then why have attempts to directly train general, and therefore highly abstract, problem-solving concepts and skills not met with hoped-for success when evaluated in terms of broad transfer? We cannot answer this question with certainty but suggest what seem to us two important reasons. First, the training goals of most problem-solving programs (e.g., Bransford & Stein, 1993; Hayes, 1981; Levine, 1988) are general skills and heuristics that can be applied across many disciplines and situations. For example, the program by Bransford and Stein (1993) is based on a very broad heuristic, IDEAL --Identify problem, Define goals, Evaluate alternatives, Act, and Look -- with a number of specific skills (i.e., memorization strategies) being nested within this general strategy. In essence, this and similar approaches attempt to create a separate subject domain made up entirely of general problem-solving abilities. However, because such training is typically very brief and because practice is provided in a wide variety of problem-solving contexts, it is unlikely that students in such courses actually achieve a stable body of knowledge with sufficient activation strength for any specific class of problem-solving situation. Thus, researchers who have attempted general thinking skills training have typically observed that, while the thinking strategies are learned and can be performed by students on demand, they are not spontaneously transferred beyond the classroom environment (e.g., Pressley, Borkowski, & O'Sullivan, 1984).

A second issue affecting the success of past efforts is that much problem-solving training has been influenced by traditional instructional design methodology, such as ISD (e.g., Gagné & Briggs, 1974) or mastery learning (e.g., Bloom, Hastings, & Madaus, 1971), which tends to offer instruction in piecemeal fashion, one or two concepts or skills at a time. The assumption underlying such approaches is that learning is "cumulative" (e.g., Gagné, 1962) in the sense that a target skill (in this case, complex problem solving) is best taught after all underlying prerequisite competencies are learned first. In such curricula, students tend to spend a great deal of instructional time focusing on specific subskills rather than on the target activity of complex problem solving.

As an alternative approach to the piecemeal training of general thinking skills, the TiPS program differs in two important ways. First, TiPS introduces general problem solving heuristics as an extension to a common body of prior knowledge, arithmetic, which people already possess to some degree and perceive as being useful and

generalizable to many everyday situations. Our approach is based on the assumption that the great many situations that already naturally activate arithmetic knowledge may also invoke other thinking skills that come to be associated with that knowledge through TiPS training. In addition, TiPS differs from traditional training in its advocacy of instructional methods based on cognitive apprenticeship (e.g., Brown, 1987). With cognitive apprenticeship, students are not first drilled on trivial problems or specific subskills until elementary ideas are mastered. Rather, they begin tackling meaningful problems very quickly, learning relevant subskills as required in the context of problem-solving activity. Participation in meaningful problem solving is made possible for novices because they share problem solving with a more advanced helping "other," a TiPS tutor. During shared problem solving, the tutor helps the student internalize critical subskills. One result of this approach is that students spend much greater amounts of time practicing the actual target performance, complex problem solving, as opposed to being drilled during much of their limited training time on isolated, hence "abstracted," subskills.

Arithmetic Story Problems as Training Context

Why choose arithmetic stories as contexts for adult training of generalized thinking skills? If general problem-solving skills are to be useful thinking tools for people, we must find a way to overcome the age-old problem of transfer. To the extent that arithmetic is used fairly often in everyday problem solving, the presence of arithmetic in a real-life problem-solving situation may have potential for serving as a sort of "trigger" that will help activate a full set of thinking skills that, through training, come to be associated with the arithmetic domain. To help enhance this "triggering effect," arithmetic training stories can be written or otherwise produced (i.e., with video in the manner of the Cognition and Technology Group at Vanderbilt, 1990) to resemble many everyday and job-related problem scenarios, providing for the possibility that training can sensitize learners to greater awareness of arithmetic in their daily lives. Consider, for example, that naval personnel on rescue missions or in combat must be able to solve complex arithmetic problems quickly with lives hanging in the balance. An example training problem might depict a petty officer, participating in a search and rescue mission at sea, being required to perform quick calculations to determine how much search area can be covered how quickly by helicopters before their fuel runs out. In addition, the sailor might be required to estimate the probable survival of victims from a sinking ship, depending on how long the search takes, water temperature, wind conditions, etc.

To the extent that such story problems bear resemblance to real-world situations, it is of serious concern that many adults have great difficulty solving them (e.g., Derry, 1989). Such concern is further justified by news reports of serious real-world errors that closely resemble the kinds of mistakes that adults frequently make on tests and in laboratory experiments with complex arithmetic story problems (Derry, 1992). For example, in Tallahassee, FL, an arithmetic error on an evaluation led to minimum security placement of a dangerous criminal who subsequently escaped and committed a serious violent crime. Finding that inexpensive training with story problems could reduce such errors and improve problem-solving efficiency would be very important.

Contexts for Studying Transfer

Our focus on transfer requires consideration of how distinctions among different problem-solving domains and situations should be viewed for purposes of both training and research. We therefore distinguish among (a) subject domains, (b) context domains, and (c) social problem-solving situations. This trichotomy is useful in that it permits us to speak of transfer between and within subject, contextual, and social domains. For example, if a sailor is taught problem solving using arithmetic word problems involving NBC defense, and if this training then enhances her ability to solve or learn to solve trigonometry problems involving NBC defense, the type of transfer exhibited is between subject domains (from arithmetic to trigonometry) but within a single context domain (NBC defense). If, on the other hand, arithmetic problem-solving skills transfer from word-problem training given in the context of NBC defense to actual grocery shopping, then transfer has occurred within subject domains (arithmetic) but between context (from NBC defense to grocery shopping) and social (from the training environment to the grocery store) domains. It follows from this analysis that the most difficult and rarest form of transfer would involve crossing context, subject, and social domains simultaneously. Such transfer would be expected only if there is useful knowledge common to all domains and if it is successfully abstracted and activated (i.e., "transferred") at appropriate times by trainees.

Our tripartite analysis will permit us to investigate and test both weak and strong forms of our training transfer hypothesis. For example, a relatively weak form of the hypothesis would predict transfer that is between social situations but within subject and context domains, or transfer that is between context domains and social situations but within subject domain. The strongest form of our hypothesis predicts transfer of training across subject, context, and social situations. Strong transfer would be exhibited if, for example, TiPS training with arithmetic problems could be shown to

enhance the problem-solving performance of electronics assembly technicians working in teams on a plant floor. It should be noted, however, that such strong forms of transfer have rarely, if ever, been demonstrated.

The TiPS Curriculum

As previously emphasized, TiPS is being designed to support particular curricular objectives and to be implemented within a training context based on a "cognitive apprenticeship" philosophical orientation (Collins, Brown, & Newman, 1989; Brown, Collins, & Duguid, 1989; Derry & Lesgold, in preparation). Correct use of TiPS depends on good understanding of these ideas. In the following sections we will describe the TiPS curriculum and how the current system, TiPS 1.0, can be used to support it. For each curriculum objective we will provide an overview, including some theory and research supporting its inclusion in TiPS training. We will then describe the cognitive apprenticeship theory that is guiding system development and that should also guide instructors who are implementing TiPS in adult learning environments. Finally we will provide some concrete suggestions on how to use TiPS in training.

The TiPS Curriculum Objectives

The following ten learning objectives are supported by TiPS.

1. Arithmetic schemas: knowledge of conceptual systems shown by cognitive research to underlie human understanding of basic mathematics.
2. Strategic planning: the ability to select and organize schemas into solution steps that achieve an overall conceptualization of and solution to a complex problem.
3. Comprehension monitoring: the tendency and ability to be aware of one's own level of understanding and to recognize when needed information is missing.
4. Systematic error-checking: the habit of checking one's problem-solving performance for avoidance of careless errors.
5. Supporting beliefs: the maintenance and use of beliefs associated with good problem solving, such as the beliefs that problem solving requires persistence and that problem solving is supposed to be challenging, not easy.
6. Selective encoding: the ability to identify important information and exclude extraneous information from the problem statement or situation.

7. Strategic search: the ability to recognize the need for and carry out searches for problem information available through common indexed sources, such as dictionaries, reference materials, transportation schedules, and manuals.
8. Strategic integration: the ability to integrate and combine information from disparate sources to achieve a unified problem representation and solution.
9. Cognitive negotiation skills: the ability to work with others to negotiate a common problem solution based on mutual understanding of the problem.
10. Multiple representational levels: The ability to construct mental problem representations at different grain sizes or levels of abstraction, an important memory-management technique.

This list of problem-solving skills and concepts represents not only a proposed set of curriculum objectives but also hypotheses about what types of training should improve problem-solving performance. Gleaned from a broad spectrum of recent cognitive and task analysis research, the TiPS curriculum does not attempt to represent all possible problem-solving subskills; rather it focuses on a few key training objectives that are recommended by cognitive research and/or theory and that should significantly improve problem-solving ability if appropriately taught and measured. Each is discussed in greater detail below.

Objective 1: Schemas of Problem Solving

Inspired by the work of Marshall and others (e.g., Carpenter, Moser, & Romberg, 1982; Marshall, in press; Marshall, Pribe, & Smith, 1987; Riley, Greeno, & Heller, 1983; Vergnaud, 1982, 1983), the schema hypothesis advances the argument that difficulties with arithmetic story problems derive largely from inadequate knowledge of basic conceptual systems, called "schemas." Schemas are conceptualized as domain-specific knowledge structures that allow patterns in problem situations to be recognized and that indicate actions associated with those patterns. External stimuli associated with problem situations are encoded in memory through a process whereby stimuli activate schemas, which attempt to map themselves onto particular aspects of the problem (e.g., Kintsch & Greeno, 1985; Marshall, in press). These mappings, when successful, constrain how each problem situation is understood, chunked, and encoded by memory.

A well-documented finding of cognitive psychology is that a major difference between experts and novices within any problem domain is that experts know many more domain schemas (e.g., Larkin, McDermott; Simon & Simon, 1978). These domain schemas are usually normative in the sense that they represent personal

encodings of the conceptual inventions of a domain's problem-solving culture that have been propagated through schooling, work, play, and other forms of social enculturation. An individual's knowledge of schemas may be explicit and conscious but, in reality, is often tacit and unconscious.

Adults in today's society may possess neither tacit nor explicit arithmetic schema knowledge because textbooks and instructional methods of the past tended to teach arithmetic operations independently of word problems and other broader conceptual contexts. Peterson, Fennema and Carpenter (1988/89) argue that such instructional approaches have not promoted schema acquisition. Based on their work (e.g., Carpenter & Fennema, 1993), public school texts and curricula are beginning to adopt a schema-based approach to basic mathematics.

Schema Types. Several researchers have analyzed the arithmetic schemas in arithmetic word problems and, despite differences in detail, the analyses of these researchers have notable similarities (e.g., Carpenter & Moser, 1982; Marshall, in press; Marshall, Barthuli, Brewer, & Rose, 1989; Morales, Shute & Pellegrino, 1985; Vergnaud, 1982, 1983). TiPS 1.0 is based on the theoretical system of schemas proposed by Marshall, which has been only slightly adapted for use in our research. Marshall's theory was chosen because it was designed specifically for instructing adolescents and adults. The system is based on five schemas -- change, group, compare, restate, and vary -- each representing a different basic set relationship that occurs in story problems. One-schema problems are the simplest possible form. Complex story problems are built around interrelated schemas. In combination, these five schemas characterize almost all word problems used in pre-algebra textbooks (Marshall, in press). The schema system is summarized in Appendix A, which provides a diagrammatic representation, a definition, and a simple concrete example situation for each schema. Readers not familiar with arithmetic schema theory should consult Appendix A before proceeding.

Position of Unknown and Schema Difficulty. Even when different problems represent an identical schema type, they may not be equally difficult. The wording of a problem can make it more clear or obscure (Hudson, 1980). Also (Carpenter & Moser, 1982; Morales et al, 1985), the position of the unknown can alter the difficulty of a schema type. Consider the following scenario describing a group relationship among four sets: "Mr. Green grows oranges, grapefruits, and avocados. If he has 3000 fruit trees in all and there are 2000 orange and grapefruit trees, then Mr. Green has 1000 avocado trees." This group relation can be made into three different problems depending on which of the four values is the unknown. There are significant

differences in difficulty depending on which value is unknown. For example, "How many fruit trees does he have in all?" is a less difficult question than "How many avocado trees does he have?"

The mathematical operator used to solve the problem also changes, depending on which value is unknown. To find total fruit trees one would use addition, but finding the number of avocado trees would require subtraction. Thus, schemas are not associated with particular arithmetic operations. However, the schemas of vary and restate are more often associated with multiply and divide, while change, compare, and group are mostly associated with addition and subtraction.

Schema Understanding. Marshall (1988) proposed that there are four distinct types of knowledge about schemas: feature recognition, constraint mapping, planning/goal setting, and execution. Feature recognition includes knowledge about the schema, such as its general description. Constraint mapping is knowledge of the rules about how to fit a schema to a particular type of problem situation. Planning/goal-setting involves using the schema to manage the problem solving process. Finally, execution concerns selecting and carrying out arithmetic operations. To say a student completely understands a schema is to say that the student has achieved many levels of schema knowledge.

Schemas and Complex Problems. Solving complex problems involves identifying and solving multiple schemas and chaining them together. In complex problems, most schemas contain values that must be determined from other schemas. Consider the problem, "Joe won $100 in the state lottery. He spent some of it on toys for his children. He bought a doll for Sue that cost $25 and a stuffed bear for Ellen that cost $35. How much money did he have left after he bought the toys?" This problem can be solved by using a group schema ("He bought a doll for Sue that cost $25 and a stuffed bear for Ellen that cost $35. How much money did he spend on toys?") to provide a number for a change schema ("Joe won $100 in the state lottery. He spent X on toys for his children. How much money did he have left after he bought the toys?") More complex problems, obviously, can have many schemas chained together. Helping students learn to manage this complexity is part of the strategic planning process, discussed next.

Objective 2: Strategic Planning

Story problem solving is conceptualized as an active, reciprocal process of constructing, testing, correcting, and operating on a mathematical problem model (Derry, 1989; Reusser, 1993). Although schemas are important building blocks of such

models, complex problem solving requires a mental process whereby schemas are strategically concatenated together so that a solution path is formed leading to the goal. Chained together meaningfully, schemas comprise a complex tree structure whereby data output from one schema serves as data input to another. This mathematical problem model is what students actually learn to construct when they solve and analyze problems with the TiPS system. The TiPS displays in Figure 1 (inserted as next page) and Appendix A show fully constructed mathematical problem models.

Reusser (1993) has advanced a theory suggesting that construction of mathematical problem models should be preceded by construction of another mental model of the story problem, the "episodic situational model." Consider the "sinking ship problem" on the TiPS screen of Figure 1.

In a damage control drill, a dewatering operation will be conducted based on the flooding of a damaged compartment.

Assume a situation in which the ship is taking on water at the rate of 400 gallons per minute and will sink after taking on 80 tons. There is one P-250 pump operational, which directs discharge of water overboard at a low pressure rate of approximately 300 gallons per minute. There is also a small hand-operated pump capable of pumping 65 gallons per minute. (Note, 1 gal weighs 8 1/3 lb., and 1 tn is 2,000 lb.)

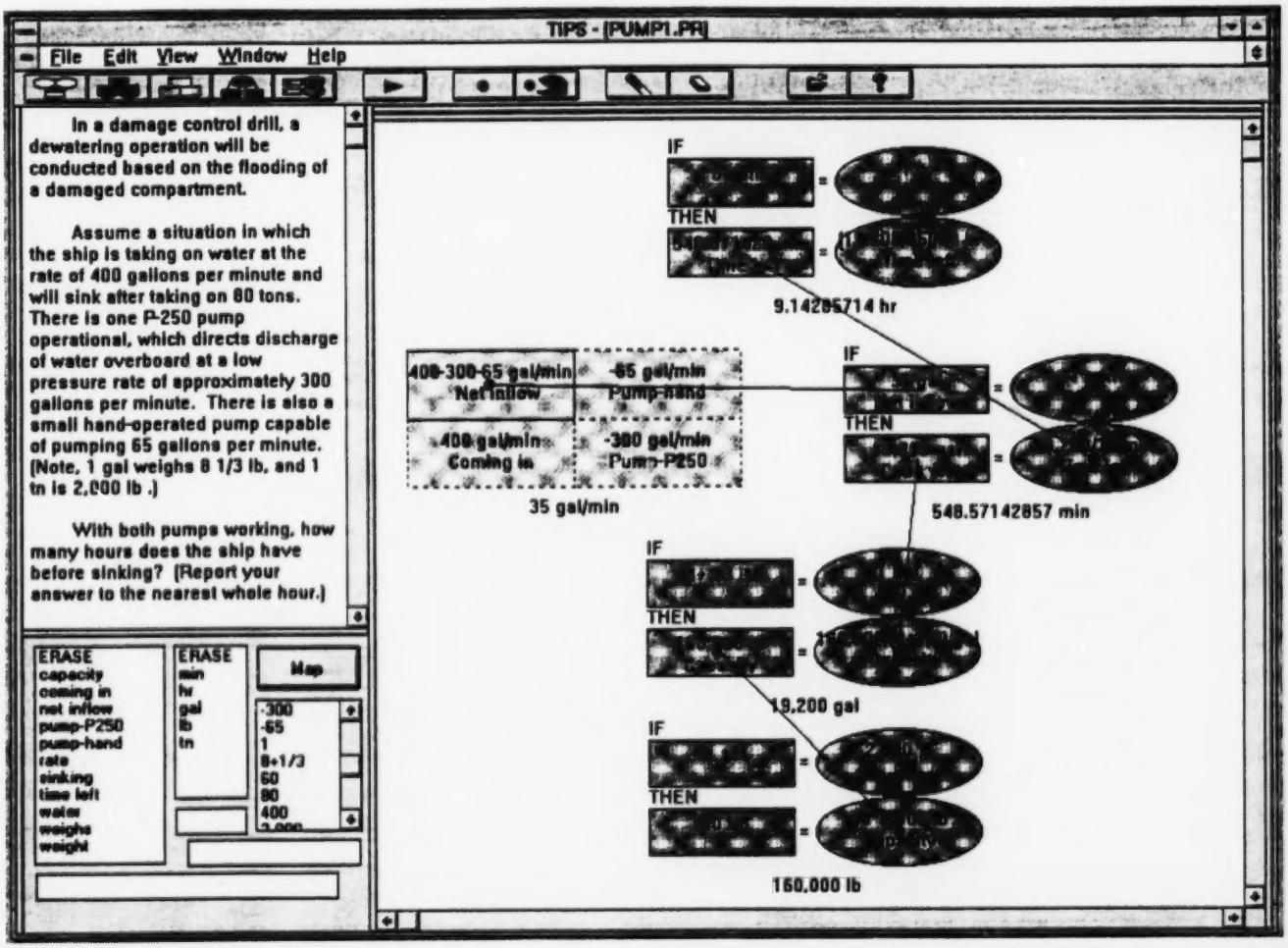
With both pumps working, how many hours does the ship have before sinking?

Before constructing the mathematical problem model shown on the screen, the problem solver, according to Reusser's theory, would have to fully understand the problem's underlying nonmathematical situation, which can be paraphrased as follows: *Some damage has occurred causing a ship to sink. It is both taking on and pumping out water at rates stated in gallons per minute. It is taking on more water than it is pumping, so will eventually sink. It will sink when it takes on 80 tons of water.* Once the situation is understood, a mathematical model can be constructed.

Working Backward. One heuristic strategy for constructing a mathematical problem model from the situation model involves looking at the problem goal (the question) and working backward systematically and logically from this goal to the givens of the problem.

The goal of the example problem above is to find how many hours it will take for the ship to sink. Reasoning from the episodic story model, the problem solver can

Figure 1. TiPS Screen Display


TIPS - [PUMP1.PR]
File Edit View Window Help
In a damage control drill, a dewatering operation will be conducted based on the flooding of a damaged compartment.
Assume a situation in which the ship is taking on water at the rate of 400 gallons per minute and will sink after taking on 80 tons. There is one P-250 pump operational, which directs discharge of water overboard at a low pressure rate of approximately 300 gallons per minute. There is also a small hand-operated pump capable of pumping 65 gallons per minute. (Note, 1 gal weighs 8 1/3 lb, and 1 tn is 2,000 lb .)
With both pumps working, how many hours does the ship have before sinking? (Report your answer to the nearest whole hour.)
ERASE
capacity
coming in
net inflow
pump-P250
pump-hand
rate
sinking
time left
water
weight
ERASE
min
hr
gal
lb
tn
Map
-300
-65
1
8+1/3
60
80
400
IF
THEN
9.14285714 hr
400-300-65 gal/min
Net inflow
-65 gal/min
Pump-hand
400 gal/min
Coming in
-300 gal/min
Pump-P250
35 gal/min
548.57142857 min
19,200 gal
160,000 lb

BEST COPY AVAILABLE

easily infer that this answer can be obtained by determining how many hours will elapse until the ship takes on 80 tons of water. But this answer cannot be directly calculated with the given information. However, the problem solver knows that problem data are reported for minutes and that minutes can be converted to hours, so a schema that can compute the final conversion is:

IF

60 min = 1 hr

THEN

? time left in min (subgoal) = ?time left in hrs (goal)

But as the number of minutes to sinking is not given and must be calculated, the intermediate subgoal problem becomes finding a method for computing the number of minutes to sinking. Here the problem solver might infer that if she can determine how many tons the ship takes on in one minute, then she can use this information in a vary schema to determine how many minutes will be required to take on 80 tons. However, since it is much easier to compute how many gallons are taken on in one minute, the problem solver migh decide to think and compute in gallons rather than tons:

IF

? net gal inflow per min (new subgoal) = 1 min

THEN

? gal capacity (new subgoal) = ? time left in min (subgoal)

Working backward from the above schema, the problem solver would have to set up schemas to calculate the new subgoals: one to determine the net gal inflow per minute and another to determine the "gal capacity," or the number of gallons in 80 tons. The problem solver of Figure 1 used a group schema to calculate net gal inflow, and a vary schema to calculate gal capacity. Once these were determined, it was possible to compute the original subgoal (time left in min), which then enabled computation of the goal, time left in hours. Thus, a problem goal can be obtained by continuing to work backward from the unknowns until a computable schema is constructed and a chain of computations can begin.

The summary above suggests how the heuristic strategy of systematically working backward from the problem goal might help a student translate from an episodic situational model to the complex mathematical problem model that underlies

mathematical understanding and supplies an answer to the problem. This working backward heuristic strategy is not always easy to apply, but it can be taught to students and employed with TiPS. It is summarized as follows:

1. Understand the episodic situation of the problem.
2. Determine the problem goal--what question is being asked.
3. Determine what type of schema will produce the goal then select and position that schema diagram on the screen.
4. If computations cannot be performed with the final schema due to missing values, set up needed subgoal schemas that will compute the missing values needed to compute the goal schema.
5. If the subgoal values of the new schema(s) cannot be computed directly, then set up the sub-subgoal schema(s) that will compute missing values needed for the subgoal schema(s).
6. Continue working backward in this manner until given values are found that enable computation of needed subgoals. This process will have created a tree representing the problem model.
7. After the tree has been constructed, backtrack from the givens by calculating through the tree to fill in the values and complete the problem model.

Working Forward. Another problem solving strategy often used by good problem solvers is working forward from the givens of a problem. This strategy will be the natural tendency for most students in most situations.

With working forward, the problem solver may start computing before fully understanding the situational problem model. For example, the student might see the words *gallons* and *tons* in the first sentence and decide before reading further to go ahead and determine how many gallons there are in 80 tons. Then, still not having a definite goal in mind, the student might use information that the ship is taking on 400 gallons of water per minute, and pumping out 365 gallons per minute, to compute a net inflow of 35 gallons per minute. Having this information available, that the sinking weight of 80 tons is 19,200 gallons and that the ship is taking on 35 gallons per minute, the student can now read further and realize that dividing 19,200 gallons by 35 gallons will give the number of minutes until sinking, which is easily convertible to hours to provide the answer. With this approach, the correct answer is derived, not by strategic planning, but by working computable chunks of the problem as they are read and understood. This process appears to simplify the data so that critical connecting insights

can be seen more easily. Our research has shown that much complex problem solving is accomplished by this data-driven strategy, which we have called "working forward" (e.g., Derry, 1989). A shortcoming of this strategy is that students attend to extraneous information which may lead them to perform unnecessary calculations and even mislead and confuse them.

Early expert-novice studies indicated that domain experts usually work forward from givens, as described above, in a rapid, data-driven manner. The seeming automaticity and forward nature of expert solution processes was explained by the fact that experts, compared to novices, have more and better (more automatic) schema knowledge. Also, through many hours of practice, experts have combined their basic schema knowledge to acquire complex schemas that incorporate ready-made plans for many types of complex problems. Novices, on the other hand, lack such structures and thus rely on such strategies as working backward from the goal (Larkin, McDermott, Simon & Simon, 1980).

However, Simon and Simon (1978) note that, in expert-novice studies, experts are usually given easy problems, because novices must be able to solve them too. It is therefore not surprising, when experts are given easy problems, that they work forward. But Larkin et al. (1980a) found that experts used the working-forward strategy only with easy problems. This suggests that, even for experts, knowledge of strategies such as working backward can be helpful for complex, nonroutine problems. It also suggests that working backward might be useful as a fall-back strategy, which can be engaged when the more natural working forward approach seems to be failing.

Research by Malin (1979) indicated that problem solvers naturally use a mixture of construction strategies (she called them "search heuristics"): backward, forward, and mixed. Her work also indicated that the efficiency of a strategy depends on such factors as problem structure, and that training in construction strategies can improve problem-solving performance. In view of these findings, the TiPS training program focuses on increasing students' explicit metacognitive awareness that model construction can follow a forward, a backward, or a mixed strategy. We do not believe that current research warrants tying particular construction strategies to particular problem features. We do believe that the strategies of working forward, working backward, and combining working forward with working backward should be explicitly taught; these strategies are tools that can be used as needed by students to help them think about complex problems while trying to understand and solve them.

Objectives 3, 4 and 5: Metacognition

In agreement with Flavell (1979, 1981), Schoenfeld (1985, 1987), Salomon and Perkins (1989), Garofalo and Lester (1985), and others, we use the term metacognition to refer to both problem solvers' self-intelligence and their ability to regulate and control their own thinking. Persons possessing intelligence about their own operation can consider the following kinds of questions: What strategies do I know and which should I try for this situation? What are my strengths and weaknesses, limits and error propensities? Persons possessing the ability to regulate their own thinking will raise issues like the following during a problem-solving episode: Am I making progress? Do I understand this? Do I have all the information I need, or is something missing? What kind of error did I make, and how can I repair and continue? The TiPS curriculum objectives of monitoring, error checking, and acquiring supportive belief systems represent our attempt to foster development of this form of problem-solving intelligence.

An individual's ability to conduct on-line monitoring of the problem-solving process is an important aspect of metacognitive intelligence. Almost daily, news reports attest to the fact that people frequently make mistakes that waste and destroy the resources of our world. Facts behind these headlines often show that much of this human error is caused, not by lack of fundamental procedural or conceptual knowledge, but by careless slips, lack of effort, failure to plan, or loss of attention. The TiPS curriculum addresses this issue through training in systematic self-checking, which aims to habituate behaviors that decrease careless errors.

TiPS training also addresses comprehension monitoring, which aims to enhance awareness and remediation of comprehension failure. A long established fact of reading research is that good readers are distinguished from poor by their comprehension monitoring skills (Brown, 1978). Similar results have been obtained with problem-solving research. For example, Schoenfeld (1983) has observed that better problem solvers behave as if equipped with ever-present monitors operating just below the surface of consciousness. When the domain is familiar and no errors occur, problem solving proceeds on automatic pilot. But when the problem posed is in some way unfamiliar, difficult, or ambiguous, the proficient problem solver behaves as though a hidden monitor is active, which alerts the processor whenever the process may be going awry. The less proficient problem solver seems oblivious to warnings, such as unfamiliar concepts, unreasonably complicated reasoning, or long strings of calculations that do not converge on a goal. When such signals occur, the expert automatically interrupts processing and seeks clarification or reexamines the problem-solving strategy.

The poor problem solver is more likely to conduct a "wild goose chase," without awareness that difficulties are occurring.

In addition, metacognitive knowledge includes beliefs, described by Norman (1981) as a type of subjective knowledge that has important influences on monitoring behavior. According to Schoenfeld (1985), the belief system involved in mathematical problem solving refers to one's problem-solving world view, which includes personal attribution and affective variables. One's perceived personal control, for example, has effect on one's problem-solving performance. For example, people who believe that different strategies will pay off are more likely to seek them than people who do not (Bandura, 1982). Or, people who believe that they seldom make careless errors are not likely to engage in systematic self-checking. In addition to personal attribution and affective variables, there are specific beliefs about mathematics problem solving that affect motivation and persistence. For example, a student who believes that it is necessary to know a correct formula to solve a word problem will make little effort to solve any problem that does not map onto a formula.

In accordance with the hypothesis that developing metacognitive skills and beliefs will improve problem solving (e.g., Pressley et al., 1986; Schoenfeld, 1987), the TiPS curriculum includes training designed to increase comprehension monitoring and error checking, as well as instill facilitative beliefs about the problem-solving process.

Objectives 6, 7, and 8: The Problem of Inconsiderate Text

Unlike word problems found in textbooks, problems of the real world do not present themselves in neatly written paragraphs containing all information necessary for solution. In a sense, solving problems in real life is like trying to construct a model of comprehension from extremely "inconsiderate text." In reality, problems often arise ambiguously, with relevant problem data missing or intertwined with distracting and/or irrelevant stimuli. That such environmental factors do create significant difficulties is indicated by research demonstrating that students often cannot identify their own knowledge deficits, and that they have difficulty detecting contradictory information, identifying missing data that are necessary for a solution, and discriminating superfluous from necessary information (e.g., Graesser & Person, 1994).

In addition to filtering out irrelevant data and focusing attention on relevant information, real-life problem solvers must frequently search for and find sources for information that may be missing entirely, not only from their own knowledge bases, but also from the surrounding problem environment. In such cases, problem solvers must

bring together disparate sources of knowledge and integrate them together in construction of a problem model. We characterize these skills as *selective encoding, strategic search,* and *selective integration.* Sternberg (1988) incorporated similar objectives into his intelligence training programs. Such skills, he argued, are key components of general problem solving abilities known as insightful learning and inductive reasoning.

For the reasons just described, some training exercises in TiPS are story problems presented with inconsiderate text. We identify four characteristics of problem presentation that are deliberately incorporated, in different combinations, into these "inconsiderate" story problems: (1) irrelevant, distracting stimuli are present; (2) the structure of the problem environment does not draw attention to relevant data; (3) needed information elements are missing from the problem presentation entirely, requiring the problem solver to recognize its absence and deliberately engage in questioning or search; and (4) problem solution requires students to integrate ideas and information from multiple sources. More difficult inconsiderate problems possess all four characteristics, whereas easier exercises contain only one or two features.

Objective 9: Problem Solving as a Social Process

Our concern with transfer to real-world applications has led us to examine implications of the fact that real-world problem solving is both a social and an individual activity. To this end, we begin with a theoretical perspective elaborated by Derry (1992), which permits us to define both individual and social problem solving as special cases of distributed cognition. The central idea of this model is that human * human interactions of any complexity can appropriately be conceptualized as the output of some cognitive information-processing system *P*. For example, the behavior and thoughts of a person solving an arithmetic story problem are the outputs of a cognitive system whose schema, strategy, and metacognitive knowledge work together. However, if two students join together to solve an arithmetic story problem, then information processing system *P* consists of two separate cognitive systems that must coordinate their knowledge to achieve a goal. To the extent that both students possess and activate similar schema, strategy, and metacognitive knowledge, a shared problem model can easily evolve. But if knowledge or viewpoints differ, a joint problem solution must be negotiated and cognitive conflict may arise (e.g., Derry & Tookey, 1994; Teasley, 1992).

Clements and Nastasi (1988, 1992) drew a useful distinction between cognitive and social negotiation. If a problem solution is derived through purely social

negotiation, then a mutual problem model may not be achieved. For example, one person may socially dominate the problem-solving group and impose a personal problem model that is not shared. Others may capitulate or approve without understanding. By contrast, cognitive negotiation implies that problem solvers converse to achieve a solution based on a shared problem model, although our theory does not suppose that two personal understandings will every be exactly alike. From a cognitive theoretical viewpoint, relatively little is known about teaching effective processes of cognitive and social negotiation. However, we assume that, like other types of problem-solving skills, skills for cognitive and social negotiation can be acquired through training.

Although social negotiation strategies may often lead to effective problem solving, we argue that cognitive (as opposed to social) negotiation is usually preferable in training because it builds mutual understanding and promotes knowledge acquisition. In support of this supposition that cognitive negotiation leads to knowledge acquisition, consider a possible situation involving pairs of problem solvers. The two problem solvers together possess the sum of all knowledge used to construct a mutual problem model and achieve solution, but before the problem model is constructed, neither problem solver individually held the knowledge that was employed to achieve solution. However, by achieving a mutual understanding of the completed problem model through negotiation, both parties must come to possess the missing model pieces; hence, learning would occur for both individuals. The basic idea of this example can easily be extended to the case in which one party (the "tutor") possesses all required knowledge but the other (the "student") does not, and to the case where neither party possesses knowledge in a form that can be directly applied to construction of the problem model. The last case in particular presents an especially difficult cognitive negotiation challenge and is more likely to require very refined negotiation strategies.

The conclusion of our analysis for developing a problem-solving training program is that, in addition to helping individuals develop schemas, strategies, and other types of knowledge that can serve them when they are working alone, we must also help students acquire cognitive negotiation strategies involved in group problem solving.

Objective 10: Memory Management and Multiple Representations

It is proposed that another important aspect of problem solving is the ability to form and operate on multiple problem representations that can model a problem at different levels of abstraction. A well-documented aspect of cognitive processing is that the span of human attention (working memory) is limited. In general, problem solvers

are able to consciously process only between five and nine chunks of information at a time (e.g., Miller, 1956). If one views problem comprehension as the process of constructing a problem model, it follows that, for complex problems, the problem model that one constructs during comprehension and problem solving will often exceed human attentional capacity. This difficulty is a significant one, for while problem understanding depends on perception of an entire problem model, working memory limitations do not allow humans to perceive complex problems in their entirety all at once.

We know that humans have evolved various methods for increasing their attentional limitations. One well-documented method is the acquisition of domain expertise. Research indicates that experts, who have completed hundreds of hours of practice within a domain, evolve complex information structures (schemas in our terminology) associated with highly automatic processes (e.g., Anderson, 1993). Expert schemas process information so efficiently that they require very little attentional capacity for execution (e.g., Schneider, 1985). Thus, in an expert's problem model, each information chunk subsumes much more knowledge and requires much less attentional capacity than do the information chunks in a novice's model. In sum, one way that humans extend their limited attentional capacity is to intensively practice domain problem solving and acquire domain expertise in the form of schemas.

However, many problem situations and their respective models are so complex and require so many information chunks that they cannot be held in attention all at once, even by experts. Also, even experts occasionally want to consciously "unpack" their unconscious, automated schemas and focus their attention on particular problem details at one time, but focus on a more holistic overview of the problem at another. For example, an expert helping a student to solve a problem might stop to focus on the details of how a particular part of the problem is conceptualized and processed. This observation leads us to believe that an important part of expertise is the ability to flexibly view problems at multiple levels of abstraction.

We hypothesize, then, that good problem solvers construct and work with multiple problem representations that enable them to view and understand complex problem situations at various grain sizes representing different levels of detail, abstraction, and inclusiveness. For example, a grain size representing a substantial level of detail but low level of inclusiveness would be the level of the specific arithmetic schema, which roughly represents one conceptual idea or step in the problem-solving process. The largest grain size of interest would be an abstract representation of the whole or evolving problem model, a map-like structure outlining how the smaller

conceptual chunks fit together. This type of representation is highly inclusive but very abstract, lacking in detail. This is not to say, however, that problem solvers construct and coordinate separate, nonintegrated representations. Rather, we envision coordinated complex problem models that can be attended to and understood by the problem solver at different grain sizes representing different levels of abstraction and inclusiveness.

The implication of this analysis fot TiPS training is as follows: Training should proffer multiple representational schemes, encouraging students to develop the skill of viewing problem situations from different perspectives and different levels of abstraction and grain size.

TiPS Instructional Philosophy: How Should Problem Solving Be Taught?

Having given an overview of TiPS 1.0 and the several curriculum objectives it is designed to support, we now address the issue of what form TiPS instruction should take.

Cognitive Apprenticeship

The TiPS System is designed to support instructors and instructional facilities that embrace a cognitive apprenticeship philosophy of training (e.g., Collins, Brown, & Newman, 1989; Collins & Brown, 1988; Katz & Lesgold, in press; Schoenfeld, 1987). The cognitive apprenticeship model is here offered as a prescriptive recommendation for how TiPS tutoring should ideally take place. Research has shown that not all experienced tutors naturally use this model, although some do (McArthur, Stasz, & Zmuidzinas, 1990).

Cognitive apprenticeship is an instructional approach that enables students to participate in and learn from meaningful problem solving long before they possess the intellectual abilities to solve complex problems independently. The approach requires that tutors, whether human or machine, share problem-solving responsibilities with students so that they can help direct students' problem solving and model expert thinking processes as problem solving proceeds. The tutor provides overall direction and encouragement, assuming that portion of the problem-solving job that is currently too advanced for the student to manage. Support from the tutor, or "scaffolding" (e.g., Collins et al., 1989; Palincsar, 1986), helps students succeed with the problem while freeing their memory resources to focus on particular target subskills that represent the learning objectives for a particular problem-solving session.

Cognitive apprenticeship represents the form of learning that takes place naturally in many social environments. It is believed by many researchers to be a necessary part of intellectual development (e.g., Brown, 1987; Lave & Wenger, 1991). For example, Vygotsky (1978) argued that the development of self-regulatory, or metacognitive, abilities depends on opportunity to participate in social processes that gradually transfer important forms of cognitive management from mentors to students. Because self-regulation is acquired by a process that involves first experiencing "other-regulation" (Brown, 1987), Vygotsky would likely argue that there are critical aspects of problem-solving intelligence that can be acquired only through cognitive apprenticeship.

With cognitive apprenticeship, students can begin working with meaningfully complex problems right away because they are provided with the tutorial scaffolding that enables them to participate and succeed with complex problem solving. Problem-solving sessions may focus on particular target skills as instructional objectives, but tutorial assistance is provided for other necessary subskills that are not yet in the student's repertoire. For example, if the target objective is the skill of identifying irrelevant information and the problem requires also that the student supply missing information by reading a graph, a tutor might explicitly direct retrieval of the graph information but ask the student to take greater responsibility for identifying extraneous information. Thus, even before students are fully capable, they are solving nontrivial problems and acquiring target skills within contexts resembling situations in which the to-be-transferred skills must later be spontaneously recalled and used.

In tutoring a student in accordance with the cognitive apprenticeship model, the correct amount of scaffolding is the amount that enables the student to perform at the top of his or her "zone of proximal development" (Vygotsky, 1978), the best level of performance that can currently be obtained with some tutorial support. Thus, the tutor must structure the selection of problems so that problems received are challenging beyond what the student can currently accomplish alone but not too difficult for the student to accomplish with some moderate degree of struggle and support. Although the phrase "moderate degree of struggle and support" is imprecise, a tutor can typically judge what is reasonable by watching for affective signs, such as motivation or frustration. However, as the student's problem-solving abilities improve, the tutor should gradually fade support, encouraging independent thinking. When students are able to perform previously supported problem-solving exercises without help from their tutor, they have progressed to the top of their zone of proximal development. This in turn expands their current zone of proximal development, requiring that the tutor select

BEST COPY AVAILABLE

more challenging exercises and supply the required scaffolding for the new, more difficult problems.

One way knowledge is passed from mentors to students during problem solving is through introduction and use of cognitive tools. Cognitive tools comprise the conceptual "bag of tricks" that the expert knows and uses to support problem solving. For example, the skills and ideas (i.e., schemas, strategies, etc.) that are provided by tutors as TiPS curriculum objectives can be regarded as tools. Tools also are found in the objects provided by the learning environment that permit students to incorporate new methods and symbols, which otherwise would be unavailable, into their problem-solving activity. For example, in TiPS, the cognitive tools include objects provided by the graphics interface. The TiPS tools are designed to encourage schema-based and strategic conceptualizations of problem solving. Like mentors, such instructional tools can also serve scaffolding functions, allowing students to perform tasks that they might be unable to perform if the tools were not available.

The recommended cognitive apprenticeship method for TiPS instructors is based on a three-stage model suggested by Brown (1987), which specifies different roles for the tutor at each stage. In the first stage, the tutor focuses on target problem-solving skills by modeling their use in the context of actual problem solving. The tutor thinks aloud, making public the target cognitive processes. Stage two is a complex activity whereby the tutor and student work together on joint problems. Metacognitive processes should continue to be made public through dialogue, although the tutor guides and prompts only when the student falters or makes a query. In the final stage, the tutor cedes control to the student, functioning primarily as a supportive, empathetic audience. These three processes are widely known as *modeling*, *scaffolding*, and *fading*, and they recur again and again, pushing the student's zone of proximal development forward until the student has completed the training program.

Tutoring Styles

Whether tutorial mentoring is provided by human instructors or by the system itself, the style of the tutor's interaction with students can vary widely, ranging from the totally unobtrusive tutoring style, where the tutor intervenes only at the student's request for help, to a highly controlling style, where the tutor intervenes with every error so that the student is forced to stay on an expert path. The work of Tobin and others (e.g., Tobin, 1990, April) suggests that metaphors provide a profitable and convenient way of describing and thinking about different kinds of styles. Following Tobin, we use metaphors not only to help characterize and test different experimental forms of

cognitive apprenticeship tutoring, but also to help train human tutors working with the TiPS program.

As part of their training, human tutors in TiPS experimental programs are asked to examine, develop, and alter the metaphors that describe how they interact with students at different points in the 3-stage cognitive apprenticeship model used in TiPS. We begin this phase of training with a discussion of the metaphors that Lepper and Chabay (1988) have used to characterize the seven different styles identified in their observations of human tutors: coach, cheerleader, collaborator, competitor, drill sergeant, lecturer, and Socratic teacher. Some of these metaphors are included in the following list, which associates each of the three cognitive apprenticeship stages with metaphorical tutoring styles that are appropriate for each stage:

I. Metaphors for Modeling Stage
   - Demonstrator
   - Transparent expert system (see how I think; ask me how I think)

II. Metaphors for Scaffolding Stage
   - Collaborator
   - Advisor
   - Socratic teacher
   - Guide

III. Metaphors for Fading Stage
   - Coach
   - Cheerleader

Achieving a productive instructional dialogue at stage 2 (scaffolding) is possibly the most challenging aspect of cognitive-apprenticeship tutoring. As this stage is much like traditional one-on-one tutoring, some useful ideas are provided by cognitive research on how expert human tutors and teachers control the timing, content, and style of helping interventions (e.g., Evens, Michael, & Rovick, 1991; Fox, 1993; Leinhardt & Ohlsson, 1989; Leinhardt & Greeno, 1986; Lepper & Chabay, 1988; Lepper,Woolverton, Mumme, & Gurtner, 1993; McArthur et al., 1990). A few findings are consistent across studies. For example, studies show that tutors follow a loosely ordered curriculum script and possess various strategies and activities for teaching material, but that they do not seem to base their tutorial decisions on deep understandings of student knowledge states or specific reasons why a student may have difficulty. But even though good tutors may not understand the psychology of student

difficulties, they do respond to them in various ways. In general, expert human tutors tend neither to wait for students to request assistance nor to interrupt to deliver unnecessary minilectures. Rather, experienced tutors typically intervene in response to indicators that the student is experiencing difficulty with the problem-solving process. These interventions may take a wide range of forms, including

- using questions to direct or provide hints (e.g., Fox, 1993; Graesser & Person, 1994),
- working out shared meaning of ambiguous statements (Fox, 1993),
- giving general correction (i.e., "Think about it," "Watch") but withholding explicit correction until student has time to initiate correction on his or her own (Fox, 1993), and
- giving task management hints or remedial help by discussing or naming relevant ideas or procedures or suggesting alternative pathways (McArthur et al, 1990).

Another important aspect of what good tutors do is task management, which involves not only scripting and pacing the curriculum, but also framing tutorial sessions with metalevel dialogue. Experienced tutors

- discuss goals and negotiate how the tutorial will proceed (Fox, 1993),
- select tasks that aren't too easy or too difficult (Fox, 1993),
- introduce and wrap up problem-solving tasks with framing dialogue (McArthur et al., 1990), and
- control size of steps or conceptual chunks on which the student focuses (McArthur et al., 1990).

A number of findings about expert tutoring are not consistent across studies. For example, Lepper et al. (1993) reported that expert tutors constantly monitored the affective and emotional states of older children and used specific strategies for maintaining and promoting motivation, while McArthur et al. (1990) found that tutors gave little consideration to motivating high-school students, although their responses to students often did have motivating effects. However, many of the same tutoring techniques were observed by both research groups, although they were differently interpreted. For example, in both studies tutors frequently made vague, nonspecific responses to errors: "Hummmm, are you sure?" "Are you satisfied with that move?" "What is your subgoal here?" Both sets of tutors challenged subjects ("Are you ready for a difficult problem?"), hedged their feedback when a student was wrong ("That was

a super effort, except you just missed the answer at little."), and asked leading questions ("Why do you think you missed that?").

Cognitive Apprenticeship Through Worked Examples

A convenient and simple way for instructors to implement cognitive apprenticeship with TiPS is to tutor with think-aloud worked examples. A think-aloud worked-example routine has two phases, demonstration and problem solving. In the demonstration phase, the student observes while the tutor solves a problem, thinking aloud to make underlying cognitive processes explicit. Since the instructional session should have specific skills objectives, those particular skills are emphasized in the demonstration and think-aloud dialogue. For example, if the tutoring session is focusing on the skill of comprehension monitoring, then the tutor's worked example demonstration might involve working a problem for the student while thinking aloud about comprehension monitoring processes.

This demonstration phase is followed by a problem-solving phase wherein the student or student group is given one or more problems to work, similar to the one demonstrated by the tutor. The student or group is asked to work an example, trying to model the methods and behavior just demonstrated by the tutor. If the tutor wishes, students may be asked to think aloud about their own problem-solving processes, as this will help make their thinking processes public so they can be guided and corrected by the tutor. During this phase, the tutor provides scaffolding and fading as necessary, directing primary emphasis during the session on the target skills for that session.

While learning through problem solving, users should be encouraged to look back and reflect on the worked example that has illustrated appropriate use of the target problem-solving skills. This "looking back" facility is possible on TiPS through the multiple windowing and recording capabilities of the system. On TiPS, the demonstration phase of worked-examples routines can be carried out by human tutors and captured by recordings that permit playback of both the voice and screen movements that the tutor demonstrated. Multiple windowing allows users to solve their own problem in one window, while periodically referring back to an expert worked example, which can be played and replayed in another window.

The TiPS recording feature also makes it possible for students to access and study think-aloud worked examples even in the absence of a human tutor. Using the recording facility, a researcher or instructor can carefully design permanent dynamic worked examples and enter them into the system. When any user (student or instructor) solves a problem on TiPS, the system creates and saves a "recording" of the user's

work. We have designed this recording so that it can not only be played back by the user but also be edited. An instructor or researcher can thus use this facility to create worked-examples instruction. For instance, an instructor wishing to teach a working backward strategy could create a model solution that uses the strategy, then play the session and edit it, adding dialogue that describes expert thinking. This record can then be saved and indexed by the system so that a student or tutor could request an expert worked example that illustrates working backward.

Instructional methodologies based on worked examples have been the focus of much recent research and appear to have great promise. Researchers (e.g., Cooper & Sweller, 1987; Sweller & Cooper, 1985; Ward & Sweller, 1990) have conducted studies using worked examples in algebra instruction to promote schema acquisition. They have observed that the ease of learning schemas through problem solving is inversely related to the amount of effortful exploration, or "search," required for reaching solution on practice problems. Their research indicates that problem-solving search is not necessary for schema acquisition and may even be a detriment to schema learning. They have proposed using worked examples as an alternative to instruction based on conventional problem-solving practice, as worked examples both minimize the need for unproductive search and direct more attention to the concepts and ideas that students are supposed to be learning.

In general, studies with worked examples support the idea that learning and problem solving can be facilitated more by students' studying many worked examples than by their solving many problems. Moreover, students working with computers not only rely on examples but have trouble solving problems without them (Pirolli & Anderson, 1985; see also Reder, Charney, & Morgan, 1986). Also, since students prefer worked examples (Anderson, Greeno, Kline, & Neves, 1981), and since students given worked examples and problems to solve learn faster than students given instruction without examples (Zhu & Simon, 1987), adopting a worked-examples approach to problem-solving instruction seems reasonable

However, worked examples without tutorial support would probably not be sufficient for less able students. Chi, Bassok, and others (e.g., Chi & Bassok, 1988; Chi, Bassok, Lewis, Reimann, & Glaser, 1989) studied what students learned while studying physics textbook worked examples, how they learned it, and how they used what they learned to solve problems. Think-aloud protocols were collected as students studied several pages of text, and good and poor problem solvers were analyzed separately. They concluded that: (a) in applying what was learned from examples to problem solving, poor solvers referred to the examples more often than the good

students; (b) the poor students tended to do more rereading of the examples; and (c) the good students tended to do more mapping from the example to the new problem and more checking for accuracy. Good students tended to infer more knowledge in self-explanations than did poor students and were also more aware of their comprehension failures. Also, good students tended to try to understand an example by relating it to principles stated in the text. The work by Chi et al. (1989) suggests that novices and weak students are unable to effectively process worked examples to help them learn. Thus, worked-examples routines may need to supply tutorial supports for helping weaker students utilize worked examples effectively.

Worked-examples instruction has not generally been combined with cognitive apprenticeship ideas such as modeling, thinking aloud, or scaffolding. However, research findings indicate that weak students in particular might benefit from receiving worked-examples training in conjunction with apprenticeship-style tutoring. This is the approach taken in TiPS, where students receive instruction by observing an expert "think aloud" while processing example problems. After observing the worked example, students then receive apprenticeship-style tutoring while attempting to solve their own problems. We hypothesize that worked-examples routines as implemented on TiPS will be more powerful than those used in other reported studies because they combine the most important key features of two successful instructional methods, worked examples and cognitive apprenticeship.

Instructional Routines

Use of the TiPS system always involves cognitive processing of story problems. However, the nature and learning value of that processing can be altered substantially by changing the type of problem-solving activity that is performed by, shared with, or demonstrated for students. The various problem-solving activities that a tutor knows and uses in instruction are here called instructional routines. Following Leinhardt and Greeno (1986), the term *instructional routine*, or merely *routine*, will refer to a social script that organizes problem-solving activity between an instructor and one or more students. For example, TiPS worked-examples routines are generalized scripts that always involve tutorial demonstration of particular problem-solving abilities, followed by student attempts to study and model the demonstrated abilities. However, such generalized routines can have many specific variations, depending on what activities are demonstrated and modeled. For example, in one version of worked-examples instruction, the "problem" might be to develop as many different solution trees as possible. In another, the tutor might choose to demonstrate how problems are written

to represent particular schemas. In sum, while problem-solving instruction almost always involves using story problems as stimuli, those stimuli can be processed by teachers and students in many ways. Routines, then, are the instructional scripts that contain the rules for processing problems. Through frequent use, routines can become highly familiar to both tutors and students so that they can be conveniently activated by simple cues, such as, "Today, we'll do some worked examples. The goal will be to find many different trees for the same problem. Now watch me . . . . "

Research by Leinhardt and Greeno (1986) has shown that expert classroom teachers know many such instructional routines and use them frequently. A recent study of expert teachers as algebra tutors for high school students showed that tutors also call on script-like subroutines for task management and remediation (McArthur, Stasz, & Zmuidzinas, 1990). However, in virtually all studies of the tutoring process (e.g., Evens et al., 1991; Fox, 1993; Graesser & Person, 1994; McArthur et al., 1990; Putnam, 1987), instructors and students engage in only one, limited main routine: guided problem solving aimed at finding the answer to the problem. Although a great deal of excellent instruction can surely take place during such answer-driven guided problem solving, the sole use of this routine for tutoring may limit student motivation as well as the range of knowledge that can be acquired.

Potentially, tutoring routines might be varied in a number of interesting, creative and educationally effective ways. Many ideas for tutoring routines can be gleaned from recent literature advocating constructivist and dialogic teaching approaches (e.g., Carpenter & Fennema, 1993; Cunningham, 1992; Lampert, 1990; von Glasersfeld, 1990), which contains a variety of constructivist thinking activities designed to promote authentic conceptual analysis and understanding of realistic problem situations. Rather than merely solving story problems, students in constructivist classrooms create their own problems for science and math concepts, identify real-world applications of ideas from science and mathematics, critique strategies and analyses of problems provided by others, or compare and contrast the efficacy of different problem-solving approaches before actually choosing one. Such thinking activities could certainly be employed by tutors and/or tutoring systems to help students learn within the context of a cognitive apprenticeship paradigm.

Although the question of which routines are most effective is ultimately an empirical one, it seems reasonable to hypothesize that introducing a variety of different routines into tutoring sessions might enhance students' understanding and help motivate both teachers and tutors. We therefore encourage instructors using TiPS to use a variety of tutoring routines that can add interest as well as promote use of appropriate

conceptual tools, in-depth analysis, and construction of understanding. To help tutors get started, some example routines are suggested below, although the range of actual choices is not limited to these. Most of the routines suggested below are variations on the general worked-examples script.

Routines for Tutoring: Suggestions for Tutors

Schema Learning

The routines described in this section are all variations on think-aloud worked example exercises designed to promote schema learning, schema transfer, conceptual analysis of problems, and flexibility of thinking.

Schema Analysis Routine. On TiPS, the tutor begins the session by selecting a problem assumed to be at an appropriate level of challenge for the student (or student group) and then uses it to demonstrate a schema-based analysis of the problem while thinking aloud. The purpose of the analysis should be to identify what schemas can be used to represent the problem situation. During the demonstration, any student questions that arise can be answered. If the student is sufficiently advanced, the tutor might include in the demonstration a reflective discussion (thinking aloud) of whether the problem could be analyzed in different ways using other schemas. Using the TiPS recording facility, the tutor can record this session for later playback. For convenience, such recording can be done in advance of the actual tutoring session. Sample recordings designed by system developers will soon be available with TiPS.

After the demonstration, the tutor should select another suitable problem and give it to the student to analyze, instructing the student to use the same ideas and procedures illustrated in the tutor's demonstration. While the student tries to model the type of problem analysis illustrated by the tutor's demonstration, the tutor should provide scaffolding that, in addition to focusing on schema knowledge, assists the student in learning how to use a worked example as a basis for acquiring knowledge. For instance, the tutor might encourage the student to replay and study the recorded worked example as well as help the student make appropriate use of the worked example. Weaker students might not know when to look back at the worked example or what part of the example to examine, so scaffolding of these processes should be provided.

To help promote broad transfer of schema knowledge, the tutor should use this routine frequently to cover a broad range of problem situations, schemas, and schema combinations. A variation on the routine above is to act out live problem situations with

students (such as ordering pizzas) instead of selecting a problem already in the problem bank. Alternatively, the tutor can use TiPS to do schema analyses of live video clips of movies and television shows that present everyday problems. Using live, enacted problem situations should help promote broad transfer and use of schemas and schema-analysis skills,

Problem-Writing Routine. In this routine, also designed to promote schema knowledge, the tutor identifies a particular schema or schema combination and then demonstrates the creation of a story problem representing the target schemas. The tutor should think aloud, reflecting about the constraints that the selected schema(s) place on the problem-writing task. For example, the tutor might reason aloud as follows: "The change schema always involves time, so I have to have a set in my problem that changes values when some action occurs. Maybe I could have someone putting in or taking out money from a bank account." If the new problem is created using the TiPS editing features, the new problem can be entered into the system problem bank if desired. After the problem is written and entered into the system, the tutor can then demonstrate the writing and entering of a detailed answer key that explains, for future users, what schemas are found in the new problem.

Following the tutor's problem-writing demonstration, the student should be given a similar schema or schema combination and then scaffolded while writing a problem and an answer key and (if desired) entering it into the problem bank for future students to use. Seeing their work incorporated into the system can be very motivating for students, and a special file can be created to save student problems.

Routines for Strategy Learning

Strategy-Demonstration Routine. This worked-examples routine is similar to the schema-analysis routine described above except that, instead of or in addition to emphasizing thoughts about which schemas are being used for problem analysis, the tutor thinks aloud about the strategy being used to connect schemas together into an overall problem model. Basically there are three schema-connecting strategies that can be demonstrated: (a) working backward from the goal set; (b) working forward from givens until the goal set is obtained; and (c) a hybrid strategy in which subgoal sets that occur when working forward are solved for by working backward.

Following the demonstration, the tutor presents a similar problem to the student and provides scaffolding while the demonstrated strategy is attempted by the student. As emphasized in the previous section, scaffolding involves helping the student make

productive use of the just-demonstrated worked example, which can be replayed and studied as necessary.

Overcoming Difficulties by Working Backward. There are no exact rules for knowing when to apply different strategies. Typically, the natural default approach for problem solving is working forward from the givens. Students usually begin thinking about and working on a problem at a point that is most natural for them. When this initial approach leads directly and easily to solution, there is little need to consciously invoke thoughts about strategy. However, if the student does not know how to begin a problem, reaches an impasse, or becomes confused, a strategy based on identifying the current subgoal or goal set and then systematically working backward from it may help the student overcome the difficulty.

To illustrate the use of working backward for the purpose of overcoming such difficulties, the tutor needs to experience or stage some impasse in problem solving and then illustrate how it is possible to switch strategy as a way to overcome impasse. Although the student can practice the skill of working backward anytime, the real opportunity to promote transfer of this skill occurs whenever the student naturally experiences an impasse in problem solving. Therefore, after demonstrating the correct use of working backward for overcoming difficulties, the tutor must remain on the lookout during future problem solving sessions for opportunities to scaffold appropriate transfer and use of the working-backward strategy.

Routines for Training Error Checking and Comprehension Monitoring Skills

Self-Monitoring Routine. Thinking aloud, the tutor demonstrates the working of a problem, but, in the manner of Schoenfeld (e.g., 1987), deliberately stages mistakes or failures to understand the problem. Comprehension monitoring and error checking behavior that help the tutor find and repair mistakes and/or confusions are demonstrated. Subsequently, the student is given a challenging problem to solve and is instructed to think aloud, modeling the monitoring and checking behaviors just demonstrated by the tutor. The student is scaffolded as this monitoring behavior develops with subsequent example problems of increasing difficulty.

Monitoring of Others Routine. The tutor first demonstrates the desired skill of monitoring by watching and correcting a previously prepared problem-solving recording in which deliberate mistakes have been made. As the mistakes unfold during playback, the tutor should stop the recording and discuss observed mistakes with the student. Then the tutor asks the student to watch either live demonstrations by the tutor or previous recordings in which deliberate mistakes have been made. The tutor scaffolds

students as their own monitoring and error-noticing behaviors are honed. At first, the tutor shares the responsibility of helping the student identify errors and miscomprehensions, gradually fading support as the student improves.

Monitoring instruction does not necessarily have to be presented in real time. A variation on the routine above involves having students identify mistakes in completed solution trees and explanations that are not presented dynamically in real time.

A note about use and importance of monitoring routines: In a fully constructivist classroom environment, the system might be entirely "owned" and controlled by the students themselves. In such environments, students are routinely writing and entering problems into the system. The monitoring routines therefore serve the useful function of having students check and correct each others' work as it becomes a permanent part of the system.

Modeling Supportive Beliefs

For some instructional objectives, then, worked examples tutoring involves giving demonstrations that depict the difficulty and struggle that often accompany challenging problems. While struggling, the tutor should take advantage of such think-aloud sessions to express epistemological beliefs that are supportive of good problem solving. For example, while struggling with a difficult problem the tutor might say, "This problem is really hard, but I know I can do it if I persist." Or, "You aren't supposed to get these right away." Or, "This is kind of fun, don't you think?" Whenever possible thereafter, the tutor should take advantage of opportunities to prompt the students to think positively and productively about the nature of problem-solving activity. The prompts should be designed to counter typical unproductive and incorrect beliefs, such as, "To solve a problem, you need to know a formula for it."

Selective Encoding Routine

The tutor initiates a selective encoding routine by choosing items from the problem bank that contain extraneous, irrelevant information (as described in the next main section, these are located in problem folders labeled "Extra" and "Quest.").The tutor demonstrates the skill of selective encoding through worked examples, thinking aloud to show students how irrelevant information is recognized and eliminated from the solution process. Worked examples should illustrate how thinking about schemas and schema-chaining strategies, such as working backward, can help students avoid distractions caused by extraneous information. Following demonstration, the student is

given a problem with extraneous information to solve on TiPS and is scaffolded while trying to model the previously demonstrated tutor behavior of selective encoding.

Search and Integration Routine

For tutoring this set of skills, the tutor must select problems from the problem bank or another source that do not contain all information needed for problem solution (as will presently be described, these are the "Find" problems in TiPS). The tutor then demonstrates solving such problems on TiPS, thinking aloud to show explicitly how missing information is identified, how good problem solvers reason about possible sources for information, and how to look up needed information. For more advanced problem solvers, the tutor should illustrate strategic integration by selecting problems that require use of information from several different sources. Each worked example can be followed by having students solve similar problems requiring search and integration. Students' problem solving is scaffolded in the manner of cognitive apprenticeship.

Cognitive Negotiation Routines

Demonstrating Negotiation Skills. Suggested negotiation skill goals, along with brief definitions, are shown as Appendix C. One way to begin instruction in these skills is for the tutor to obtain a partner (perhaps another instructor or an advanced student) and solve a problem with that partner while demonstrating two or three specific negotiation skills. For example, the dialogue might illustrate the skills of listening, clarifying, and consensus testing. These staged demonstrations, which can be recorded on the TiPS system in advance, should be shown to students. The tutor should explain exactly what negotiation skills are being demonstrated. Then, the student should be placed in a pair problem-solving situation, either with another student or with the tutor, and be asked to model the negotiation skills previously demonstrated. The tutor provides feedback and guidance on the student performance of the skills.

Reflecting on Negotiation Skills. In this routine, the tutor asks two or more students to work together to solve a problem, then to reflect on and discuss their interaction by answering the following types of questions: Did they truly listen to one another? Did they have two competing points of view? Were both valid? Did the final solution represent a mutual understanding and agreement, or was it socially imposed? What was the nature of the role played by each person? Could the process be made more efficient?

Tutoring with the Problem and Worked-Examples Banks

Although instructor-users can develop their own problems and worked examples in TiPS, a 100-item problem "bank" is currently included in TiPS 1.0. Version 1.1, which will be released later in 1994, will in addition contain a bank of worked examples. This section is intended as a guide for helping users understand what types of problems and worked examples will be available in TiPS 1.1 and how they will be indexed for access by students and instructors. Although user sites can add to and restructure their problem and worked example banks at will, we recommend maintaining and adding to the current structure, described below.

Structure of the Problem Directory

A bank of problems is contained in the main TiPS directory (folder) titled "Problems." This Problems Directory is organized to facilitate access to particular problem types as required by TiPS instructional goals. A detailed description of the Problems Directory is given in Table 1, inserted as the following page. Its overall structure can be outlined briefly as follows:

I. The Problems Directory
  A. Problem Theme Folders
    A.1 Problem Category Folders
      A.1.1. Problem Subcategory Folders

Within the Problems folder there are two or more "theme folders," each containing problems united by a common theme or context. The title for each theme folder has two parts separated by a dot. The part before the dot abbreviates the theme, and the abbreviation following the dot indicates whether the problem deals with civilian or military subjects. For example, theme folders entitled *Pay.mil* and *Safety.mil* contain problems that deal with military topics pertaining to pay and safety, respectively. A theme folder titled *Transp.civ* contains problems that pertain to civilian transportation.

In each theme folder there are various problem-category folders representing types of problems that are useful for instruction. Problem-category folders represent groupings of problems in terms of the number of steps or schemas employed in typical solutions (although the actual number of schemas used to solve a problem may vary with different solution approaches). Problems containing extraneous information, problems lacking critical information, and high-challenge complex problems with both

TABLE 1
SUBDIRECTORIES WITHIN THEME FOLDERS OF THE PROBLEM BANK

Subdir Title	Contents	Training uses	Contents of Subdirectories
"1schema"	Problems having 1 schema and no missing or extraneous information.	Beginning schema training. Beginning error checking and comprehension monitoring training.	CH (change problems) GR (group problems) VY (vary problems) RS (restate problems) CP (compare problems)
"2-3schem"	Problems having 2-3 schemas, no missing or extraneous information.	Intermediate schema training. Beginning strategy training. Intermediate error and comprehension monitoring training. Beginning cognitive negotiation training.	Not Applicable
"3pluschm"	Problems having more than 3 schemas, no missing or extraneous information.	Advanced schema training. Intermediate strategy training. Intermediate error and comprehension monitoring training. Intermediate cognitive negotiation training.	Not Applicable
"Extra"	Problems containing extraneous information not required for solution.	All levels of selective encoding training. Also, intermediate-advanced cognitive negotiation training.	1schema (1-schema problems) 2-3schem (2-3 schemas) 3pluschm (more than 3 schemas)
"Find"	"Find" problems lack key information that must be looked up in order to achieve a solution.	All levels of strategic search training. Also intermediate-advanced cognitive negotiation training.	1schema 2-3schem 3plusschm
"Quest"	Problems in quest folders are deliberately ambiguous as they not only contain extraneous information but also require that the solver look up and integrate information from one or more sources.	Advanced training that focuses on combining strategy, selective encoding, and strategic search skills. Also advanced cognitive negotiation training.	2-3schem 3plusscshm

35b

missing and extraneous information are separated into different folders. A full description of the subdirectory structure of theme folders is given in Table 1.

When a user working on the system selects a problem exercise from a particular directory and subdirectory, that folder remains open until the user closes it. Thus, after a problem has been selected and solved and the user returns to the problem directory for a new problem, he or she will automatically be returned to the same subdirectory from which the last problem was selected. Hence, entire tutoring sessions can be organized conveniently around themes and problem types, as well as instructional objectives.

Structure of the Worked-Examples Directory

A directory of worked examples is currently under development and will be released with TiPS version 1.1. Worked examples are edited and verbally annotated problem-solving sessions that can be played to illustrate expert problem-solving performance accompanied by explanation of the specific skills in the TiPS curriculum. The worked-examples bank is organized in accordance with TiPS instructional objectives to facilitate objectives-based tutoring and self-instruction. Although this structure may undergo some revision in future, we are currently developing the bank of worked examples following the outline below.

I. Worked Examples Illustrating Schema-based Processing
    A. Single-schema problems
        A.1. Change problems
        A.2. Group problems
        A.3. Compare problems
        A.4. Vary problems
        A.5. Restate problems
    B. Problems containing 2-3 schemas
II. Worked Examples Illustrating Strategic Planning
    A. Problems containing 2-3 schemas
        A.1. Working forward
        A.2. Working backward
        A.3. A hybrid strategy
    B. Problem containing more than 3 schemas
    C. Problem containing missing data
    D. Problem containing extraneous information
    E. Quest problem

III. Worked Examples Illustrating Comprehension Monitoring and Systematic Error Checking
    A. Problem containing 2-3 schemas
    B. Problem with more than 3 schemas
    C. Problem containing extraneous information
    D. Problem with missing data
    E. Quest problem
IV. Worked Examples Illustrating Selective Encoding of Relevant Problem Data
    A. Single-schema problem
    B. Problem with more than 3 schemas
    C. Quest problem
V. Worked Examples Illustrating Search and Integration of Information from Outside Sources
    A. Nonquest single-schema problem
    B. Nonquest multischema problem
    C. Quest problem
VI. Worked Examples Illustrating conversational Skills Underlying Cognitive Negotiation between Two Problem Solvers
    A. Clarifying and explaining behaviors
    B. Questioning and listening behaviors
    C. Consensus testing and monitoring behaviors
    D. Counterexample illustrating poor cognitive negotiation skills

Tutoring with TiPS 1.0: Concluding Comments

TiPS students are expected to study worked examples and solve problems to acquire the generalized problem-solving skills of the TiPS curriculum. System developers envision that students' study and problem-solving activities on TiPS will be guided by a combination of system and human tutoring. For the current version of TiPS, the ideal arrangement is to implement TiPS in a learning center where there are knowledgeable, trained tutors who can work closely with beginning students until their problem-solving and system skills have reached a level where tutorial support can be phased out. For more advanced students, it is sufficient to have human tutors available on an "as-needed" basis to answer questions and provide other kinds of help. Centers organized in this manner follow a cognitive apprenticeship model, characterized by strongly scaffolded problem solving in early training followed by gradually fading

support in later training. As newer releases of TiPS are made available with more powerful mentoring capabilities, less time and attention will be required from human tutors, even in early stages of training, although some degree of human interactive support may always be desirable.

TiPS as a Generalized Design

Although the current focus is on arithmetic word problems, TiPS represents a very feasible instructional system design that is significantly generalizable beyond its current subject domain. TiPS problem solutions are generalized trees containing nodes that represent basic concepts, or schemas, within a domain. Solving problems on TiPS involves (1) identifying which basic schemas are present in a story problem, (2) selecting the correct schema icons from the TiPS toolbar and placing them on the computer screen, (3) mapping parts of problems onto the schema icons, and (4) strategically connecting these schemas together to form a tree representing a solution path (see screen display in Figure 1). Thus, if an instructional designer can identify what schemas and problem-solving approaches are employed by experts in *any* domain, then problem-solving capability in that subject domain might be accomplished on TiPS by adding new schema icons to the system toolbox, by writing problems for the new domain, etc. To a great extent, the background computations that support problem solving can remain the same for all domains. Moreover, many of the actual curriculum objectives are thinking skills that apply across subject domains. For example, error checking and cognitive negotiation are important skills in virtually every problem-solving environment. This is significant because it greatly increases the range of application for many of the computational techniques and instructional methods developed on this project.

References

Anderson, J. R.. (1993). *Rules of the mind.* Hillsdale, NJ: Erlbaum.

Anderson, J., Greeno, J. G., Kline, P. J., & Neves, D. M. (1981). Acquisition of problem-solving skill. In J. R. Anderson (Ed.), *Cognitive skills and their acquisition.* Hillsdale, NJ: Erlbaum.

Allen, C. L. (1991). Multimedia learning environments designed with organizing principles from non-school settings. In E. DeCorte, M. C. Linn, H. Mandl, & L. Vershaffel (Eds.) *Computer-based learning environments and problem solving* (pp. 465-483), Berlin: Springer-Verlag.

Bandura, A. (1982). Self-efficacy mechanisms in human agency. *American Psychologist, 37,* 122-147.

Bassok, M., & Holyoak, K. J. (1989). Interdomain transfer between isomorphic topics in algebra and physics. *Journal of Experimental Psychology: Learning, Memory, and Cognition. 15,* 153-166.

Bloom, B. S., Hastings, J. T., & Madaus, G. F. (Eds.), (1971). *Handbook on formative and sumative evaluation of student learning.* New York: McGraw-Hill.

Bransford, J. D. & Stein, B. S. (1993). *The ideal problem solver* (2nd ed.). New York: Freeman.

Brown, A. L. (1978). Knowing when, where and how to remember: A problem of metacognition. In R. Glaser (Ed.). *Advances in instructional psychology,* (Vol. 7, pp. 55-113). Hillsdale, NJ: Erlbaum

Brown, A. L. (1987). Metacognition, executive control, self-regulation, and other more mysterious mechanisms. In F. Weinert & R. Klewe (Eds.), *Metacognition, motivation and understanding.* Hillsdale, NJ: Erlbaum.

Brown, J. S., Collins, A., & Duguid, P. (1989). Debating the situation: A rejoinder to Palincsar and Wineburg. *Educational Researcher, 18,* 10-12.

Carpenter, T. P. & Moser, J. M. (1982). The development of addition and subtraction problem-solving skills. In T. P. Carpenter, J. M. Moser & T. A. Romberg (Eds.), *Addition and subtraction: A cognitive perspective* (pp. 9-24). Lawrence Erlbaum: Hillsdale, NJ.

Carpenter, T. P., Moser, J. M. & Romberg, T. A. (Eds.), (1982). *Addition and subtraction: A cognitive perspective.* Lawrence Erlbaum: Hillsdale, NJ.

Carpenter, T. P. & Fennema, E. (1993). Cognitively guided instruction: Building on the knowledge of students and teachers. *International Journal of Educational Research, 17,* 457-470.

Catrambone, R., & Holyoak, K. J. (1989). Overrcoming contextual limitations of problem-solving transfer. *Journal of Experimental Psychology: Learning, Memory and Cognition, 15,* 1147-1156.

Chi, M. T. H., & Bassok, M. (1988). *Learning from examples via self-explanations.* (Technical Report No. 11). University of Pittsburgh: Learning Research and Development Center. (ERIC Document Reproduction Service No. ED 296 879).

Chi, M., Bassok, M., Lewis, M. W., Reimann, P., & Glaser, R. (1989). Self-explanations: How students study and use examples in learning to solve problems. *Cognitive Science, 13,* 145-182.

Chipman, S., F., Segal, J. W., & Glaser, R. (Eds.). (1985). *Thinking and learning skills: Vol 2. Research and open questions.* Hillsdale, NJ: Erlbaum.

Clements, D. H., & Nastasi, B. K. (1988). Social and cognitive interactions in educational computer environments. *American Educational Research Journal, 25,* 87-106.

Clements, D. H. & Nastasi, B. K. (1992). The role of social interaction in the development of higher-order thinking in Logo environments. In E. DeCorte, M. C. Linn, H. Mandl, & L. Verschaffel (Eds.), *Computer-based learning environments and problem solving* (pp. 229-248). Berlin: Springer-Verlag.

Cognition and Technology Group at Vanderbilt, (1990). Anchored instruction and its relationship to situated cognition. *Educational Researcher, 6,* 2-10.

Collins, A. & Brown, J. S. (1988). The computer as a tool for learning through reflection. In H. Mandl & A. Lesgold (Eds.), *Learning issues for intelligent tutoring systems* (pp. 1 - 18). New York: Springer-Verlag.

Collins, A., Brown, J. S. & Newman, S. E., (1989). Cognitive apprenticeship: Teaching the craft of reading, writing, and mathematics. In L. Resnick (Ed.) *Knowing, learning and instruction: Essays in honor of Robert Glaser* (pp. 453-494) Hillsdale, NJ: Erlbaum.

Cooper, G., & Sweller, J. (1987). Effects of schema acquisition and rule automation on mathematical problem-solving transfer. *Journal of Educational Psychology, 79,* 347-362.

Cunningham. D. J. (1992). Beyond educational psychology: Steps toward an educational semiotic. *Educational Psychology Review, 4,* 165-194.

Derry, S. J. (1989). Strategy and expertise in word problem solving. In C. McCormick, G. Miller, & M. Pressley (Eds.), *Cognitive strategy research: From basic research to educational applications* (pp. 269 - 302). New York: Springer-Verlag.

Derry, S. J. (1992). Metacognitive models of learning and instructional sytems design. In M. Jones & P. Winne (Eds.), *Adaptive learning environments: Foundations and frontiers,* (pp.) Berlin: Springer-Verlag.

Derry, S. J., & Lesgold, A. (in preparation). Toward a situated social practice model for instructional design. To appear in D. Berliner & R. Calfee (Eds.), *Handbook of Educational Psychology.*

Derry, S. J., & Tookey, K. (1994, June). Effects of collaboration and computer tool use. In *Proceedings of the 1994 Annual Meeting of the Cognitive Science Society, Atlanta, GA.* Hillsdale, NJ: Erlbaum.

Elshout, J. J. (1991). Formal education versus everyday learning. In E. DeCorte, M. C. Linn, H. Mandl, & L. Vershaffel (Eds.), *Computer-based learning environments and problem solving* (pp. 5-18). Berlin: Springer-Verlag.

Evens, M., Michael, J., Rovick, A. (1991, September). *Natural language interaction for an intelligent tutor.* Presented at ONR Grantee Meeting on Learning and Instruction, Center for Human-Machine Systems Research, Georgia Institute of Technology, Atlanta, GA.

Flavell, J. H. (1979). Metacognition and cognitive monitoring: A new area of psychological inquiry. *American Psychologist, 34,* 906-911.

Flavell, J. H. (1981). Cognitive monitoring. In W. P. Dickson (Ed.), *Children's oral communication skills* (pp. 35-60). New York: Academic Press.

Fox, B. A. (1993) The Human tutorial diaogue project: Issues in the design of instructional systems. Hillsdale, NJ: Erlbaum.

Gagné, R. M. (1962). Military training and principles of learning. *American Psychologist, 17,* 83-91.

Gagné, R. M., & Briggs, L. J. (1974). *Principles of instructional design.* New York: Holt, Rinehart & Winston.

Garofalo, J., & Lester, F. (1985). Metacognition, cognitive monitoring, and mathematical performance. *Journal for Research in Mathematics Education, 16,* 163-176.

Graesser, A., & Person, N. K. (1994). Question asking during tutoring. *American Educational Research Journal, 31,* 104-137.

Hayes, J. R. (1981). *The complete problem solver.* Philadelphia: Franklin Institute Press.

Hudson, T. (1980, July). Young children's difficulty with "How many more . . . than . . . are there?" questions. *Dissertation Abstracts International, 41 (01).*

Katz, S. & Lesgold, A. (1993). The role of the tutor in computer-based collaborative learning situations. In S. Lajoie & S. Derry (Eds.), *Computers as cognitive tools* (pp. 289-318). Hillsdale, NJ: Erlbaum.

Kintsch, W., & Greeno, J. G. (1985). Understanding and solving word arithmetic problems. *Psychological Review, 92,* 109-129.

Lampert, M. (1990). When the problem is not the question and the solution is not the answer: Mathematical knowing and teaching. *American Educational Research Journal, 27*, 29-64.

Larkin, J., McDermott, J., Simon, D. & Simon, H. A. (1980a). Expert novice performance in solving physics problems. *Science, 208*, 1335-1342.

Larkin, J., McDermott, J., Simon, D.P. & Simon, H.A. (1980b). Models of competence in solving physics problems. *Cognitive Science, 4*, 317-345.

Lave, J., & Wenger, E. (1991). *Situated learning: Legitimate peripheral participation.* New York: Cambridge University Press.

Leinhardt, G., & Greeno, J. G. (1986). The cognitive skill of teaching. *Journal of Educational Psychology, 78*, 75-95.

Leinhardt, G., & Ohlsson, S. (1989, April). *Tutorials on the structure of tutoring from teachers.* Paper presented at the annual meeting of the American Educational Research Association.

Lepper, M. R., & Chabay, R. W. (1988). Socializing the intelligent tutor: Bringing empathy to computer tutors. In H. Mandl & A. Lesgold (Eds.), *Learning issues for intelligent tutoring systems.* New York: Springer-Verlag.

Lepper, M. R., Woolverton, M., Mumme, D. L., & Gurtner, J. (1993). Motivational techniques of expert human tutors: Lessons for the design of computer-based tutors. In S. Lajoie & S. Derry (Eds.), *Computers as cognitive tools* (pp. 75-106). Hillsdale, NJ: Erlbaum.

Levine, M. (1988). *Effective problem solving.* Englewood Cliffs, NJ: Prentice Hall.

Malin, J. T. (1979). Information-processing load in problem solving by network research. *Journal of Experimental Psychology: Human Perception and Performance 5* , 379-390.

Marshall, S. P., Barthuli, K. E., Brewer, M. A., & Rose, F. E. (1989). *Story Problem Solver: A schema-based system of instruction* (Technical Report No. 89-01, ONR Contract N00014-85-K-0661). San Diego: San Diego State University.

Marshall, S. P., Pribe, C. A., & Smith, J. D. (1987). *Schema knowledge structures for representing and understanding arithmetic story problems* (Technical Report, ONR Contract No. N00014-85-K-0661). San Diego: San Diego State University.

Marshall, S. P. (1988, May). *Assessing schema knowledge.* (Technical Report, ONR Contract No. N-00014-K-85-0661). San Diego: Center for Research in Mathematics and Science Education.

Marshall, S. P. (in press). *Schemas in problem solving.* Cambridge Press.

McArthur, D., Stasz, C., & Zmuidzinas, M. (1990). Tutoring techniques in algebra. *Cognition and Instruction, 7*, 161-195.

User's Guide

Contents

Problem Solving with Schemas

1.1 Introduction

1.2 Schema Definitions

1.3 Examples

User-Interface Reference

2.1 Commands for Creating a Solution

2.2 Commands for Manipulating Windows

2.3 The Problem Window

2.4 File Types

2.5 Problem Files

1.1 Introduction

1.1.1 Story Problems

Story problems are abbreviated accounts of real-life problem situations. They differ from other types of stories because they contain a question (the problem), as well as most of the information needed to answer the question.

1.1.2 Schemas

TiPS (Tutoring in Problem Solving) was designed to encourage a new way of thinking about and solving story problems. This approach allows students to look for semantic structure in a problem before deciding which arithmetic operations to use in the solution. Conceptual thinking is encouraged, and computation is deemphasized.

Schemas are semantic patterns that are commonly found in arithmetic story problems. They represent types of relationships that exist among problem quantities (sets).

In TiPS, story problems are solved by identifying schemas in the problem text and then linking them into multschema structures. The student creates a solution diagram showing what schemas are used and how they are connected together. The five schemas of the TiPS system are defined and illustrated in sections 1.2 and 1.3.

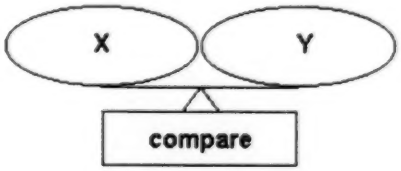
Frequently different people will see different schemas in the same problem, which may be acceptable. Often there is more than one way to think about and solve a story problem correctly.

1.2 Schema Definitions

The five schemas used in the TiPS system represent typical patterns, or set relationships, that can be found in arithmetic story problems.

1.2.1 Compare

In a compare relationship, two or more quantities are being compared to determine which is the larger or smaller (i.e. the maximum or the minimum), or to determine which is the larger or smaller and the difference between the two. Sometimes it's difficult determining what is being compared.



For example: John walked 1 mile in 15 minutes, and Joan walked 1 mile in 13 minutes. Joan walked faster than John. What is being compared here isn't stated explicitly. What is being compared is John's speed with Joan's, not number of miles or number of minutes. Since Joan walked the same distance in less time, her speed was greater than John's.

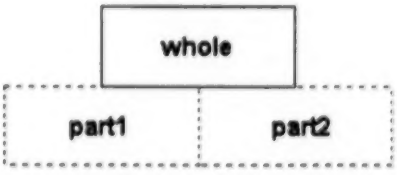
1.2.2 Change

In a change relationship, there is a physical change in something that can be measured. There's an initial quantity, some change that alters the initial quantity, and a final quantity, which is what the initial quantity becomes after the change. Time always passes during a change situation. The initial quantity and final quantity don't exist at the same time.

For example: Diane has 12 apples. To start with she only had 5, but Shingi gave her 7 more. Note, the initial quantity is 5 apples, the change was the gift of 7 more. The final quantity was the 12 apples Diane had after the change.

1.2.3 Group

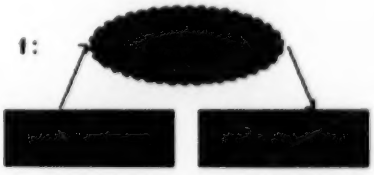
A group relationship is like a pie in that it can be viewed as a whole pie or as made up of two or more pieces. There is a quantity representing a general category, and there are two or more quantities representing specific categories that also are members of the general category. Group differs from the change situation because no time passes and no actions are taken. All the quantities in a group relationship exist together at the same time.



Example: There were 2 apples, 3 bananas, and 6 tangerines in a bowl. There were 11 pieces of fruit altogether. Here the general category is fruit, and the specific categories are apples, bananas, and tangerines.

1.2.4 Restate

In a restate relationship, you describe one quantity as a function of another quantity.

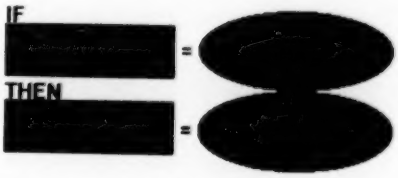


Example: Sue is 10. Her mother is 3 times Sue's age. Her mother is 30. Here the mother's age is expressed as a function of the daughter's age.

1.2.5 Vary

BEST COPY AVAILABLE

In a vary relationship, there are four quantities. Vary is a relation among two subject/object pairs. The association between subject and object is the same for both pairs. You must identify the two subject/object pairs and establish a mapping from one to the other. No action is taken and no time passes. You have to infer the association from one subject/object pair before you can map it to the other subject/object pair. The subjects and objects usually do not have the same units.



Example: The price of a certain type of electrical wiring is $2.60 per 3 feet. 9.5 feet of this wiring will cost $24.70. Here the subject/object pairs are $2.60/3ft and $24.70/9.5 ft.

1.3 Examples: Schemas in Problem Solving

Once you're able to remember and distinguish among the five story-problem schemas, you can begin thinking about and solving problems in a new way. This is done by looking for schema patterns in the problem and then structuring a solution from the schemas you found. Concentrating on arithmetic operations can be postponed until after you're comfortable with your solution structure.

1.3.1 Example Problem 1

Can you see the group and change schemas in the following problem?

Some friends would like to order out for pizza. A large pizza with three toppings costs $12. Extra cheese is $2, extra sauce 50¢; additional toppings are $1.50 each. They decide on a large pizza having 4 toppings. Sarah is collecting the money and has $10 so far. How much more is needed?

The total price of the desired pizza can be thought of as the general quantity of a group schema. The pizza's base price and additional topping would then be the specific quantities.

The goal of this problem, determining the amount of money Sarah still needs to collect, can be thought of as the change quantity of a change schema. The amount she's already collected would be the initial amount, and the total cost of the pizza would be the final amount.

A solution then, would be (note phrase-schema color associations):

Some friends would like to order out for pizza. A large pizza with three toppings costs $12. Extra cheese is $2, extra sauce 50¢, additional toppings are $1.50. They decide on a large pizza having 4 toppings. Sarah is collecting the money and has $10 so far. How much more is needed?

BEST COPY AVAILABLE

2.1 Commands for Creating a Solution

Commands for creating a solution are accessible in several locations: within the edit menu, on the toolbar, and within the quantity pane.

The commands can be organized into categories: selecting a schema, mapping phrases to schema quantities, linking schemas into solution trees, erasing, and moving solution trees.

2.1.1 Selecting

After you recognize a situation in the problem text, you can choose one of these five schemas to represent it. Clicking on the desired schema will cause the mouse cursor to take on a miniature shape of the new schema. You then indicate where it should be placed in the mapping windowpane.

For an understanding of what a schema is and how to distinguish between them, see sections 1.2 Schema Definitions and 1.3 Schema Examples.

2.1.2 Mapping

The quantity pane, in the lower left corner of the problem window, is used for constructing phrases.

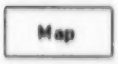
Three types of phrases can be constructed: labels, unit expressions, and arithmetic expressions. Valid examples of each are given below.

label:	`earth's surface area`
unit expr.:	`mi^2`
arith. expr.:	`3.14*7918^2`

Labels are created from terms and spaces.
eg avg. speed of bus

Unit expressions are constructed from abbreviated units and the operators: `(, ), ^, *, /.`
eg mi/hr

An arithmetic expression can be constructed from numbers and the operators `(, ), ^, *, /, +, -`. To evaluate expressions correctly, the Order of Operation rules are followed.
eg 80/1.5



After a phrase has been constructed, it can be mapped into a schema quantity.

Unmap erases a schema quantity.

Note, the map button is located in the quantity pane, but the unmap button is located on the toolbar, next to the erase button.

2.1.3 Linking

The mark/link command allows you to create multi-schema solutions to a problem, as well as identify which quantity is the answer.

To link two schemas together, activate the link command. Then, identify which quantity in the source schema is the unknown. (You do this by positioning the link cursor and pressing the left mouse button. A dot is drawn in the quantity.) Next, drag a line from the unknown to the goal quantity of another schema.

Identifying which schema quantity is the unknown, by marking it with a dot, can be a complete action in itself. (In other words, you don't always have to connect schemas when using the link command.) Note, this is how the answer quantity gets identified.

The unmark/unlink command separates two schemas and removes the dot identifying the schema's unknown quantity.

To unlink two schemas, activate the unlink command. Then, place the unlink cursor in the schema containing the dot-end of the link line.

2.1.4 Erasing

Unmap erases a schema quantity.

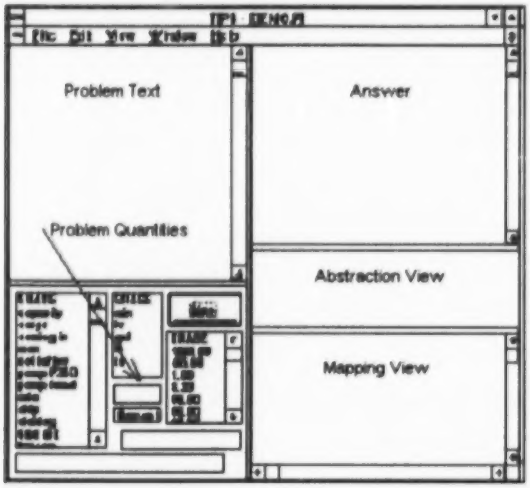
The erase command will delete an entire schema and any subordinate schemas linked to it. The most recent deletion can be retrieved with the Edit menu's Undo command, or by pressing CTRL+Z.

2.1.5 Moving

Moving a schema is easy. Just position the mouse cursor on top of the schema, press the left mouse button, and drag the schema to a new location in the mapping windowpane.

2.3 The Problem Window

After a TiPS story problem is opened and its window maximized, several windowpanes can be seen. They are shown and labeled below.



The purpose of the Quantities pane, located in the lower left corner, is to display terms, unit abbreviations, and values found in the problem text. The Answer pane displays one or more solution attempts created by instructors or students.

The Mapping view is the main workspace area where schema solutions can be created. The Abstraction view provides an alternate, less detailed view of a solution.

2.5 Problem Files

Instructors and researchers who want to create their own TiPS story problems can do so by using a text editor, like Windows' Notepad. The problem should be saved to a file having a DOS file extension of ".p" and should conform to the Problem file syntax defined in 2.5.2.

2.5.1 Problem File Example

The example below illustrates the semantics of a Problem file:

```
//Neat problem having a slightly involved algebraic
// solution, and an elegant visual solution (pie chart).

problem  "Iron Ore";

classify
  f3.difficulty
& (group,vary).schema
;

text
 "\tTwo thirds of all iron ore mined in the United"
&" States last year came from Minnesota."
&"  One sixth came from Michigan."
&"  All other states combined yielded 18 million tons."
&"\n\n\tHow many million tons, then, came from"
&" Minnesota?"
;

answer
 "\t 4(18 Mtn)  =  72 Mtn"
&"\n\n\tNote, it's probably easier to solve this"
&" problem if you can visualize it as a pie chart,"
&" rather than if you start using algebra right away."
;

values  "1/6" &"2/3" &"18";
units   "%" &"Mtn";

terms
 "USA"
&"MN"
&"MI"
&"iron ore"
&"other states"
&"total"
;
```

BEST COPY AVAILABLE

2.5.2 Problem File Syntax

The grammar below defines the syntax of a Problem file. Curly brackets indicate that the enclosed syntax element may repeat zero or more times. The syntax is free format which allows the statements in the Problem file to be spaced however it's author feels is most meaningful.

Problem =
problem *String*;
classify *Domain* {&*Domain* };
text *String* {&*String* };
answer *String* {&*String* };
values *String* {&*String* };
units *String* {&*String* };
terms *String* {&*String* };

Line commenting is supported. This type of comment ignores any text between the comment symbol ("//") and the end of the line. An example of it's usage is given above in the Iron Ore problem.

The following character constants can be inserted in strings. They are useful for formating text.

tab	\t
newline	\n
double quote	\"
backslash	\\

Appendix B - Example TiPS Word Problems

Military Contexts

Lt. Johnson is at the rifle range with an M14 rifle and is beginning to sight in. He finds that his pattern of shots is falling 18 centimeters to the right of the bull's eye. If the range is 100 meters, how many clicks must he move the sight adjustment knob? (Recall that one click on the front of an M14 rifle represents a change impact point of approximately 3 centimeters.)

A civilian visitor aboard the Nimitz estimates that the carrier is traveling about 30 miles per hour. You, a crew member, know that the Nimitz is traveling at a speed of 27 knots. The civilian wants to know how close his estimate was to the carrier's actual speed. He says he thinks his estimate was a little low. What do you tell him? (Note: 1 nautical mile = 1.152 statute miles.)

Calculate the percentage of increase in basic pay rate for an E-5 with 5-1/2 years of service who is advanced to E-6 seven months later. Report your answer to the nearest tenth digit. (Refer to figure 1-1 for pay rate information.)

Sodium hypo-chlorite (household bleach) is said to be a good biological decontaminate. When used for decontamination of clothing it should be diluted (2/3 cup of bleach in 1 gallon of water). If 4 gallons of water were required for a mixture to decontaminate a bundle of clothing, what amount of sodium hypo-chlorite should be added?

Civilian Contexts:

Eduardo is able to run a quarter mile in 1 minute, 12 seconds. Assuming he is able to keep the same pace for a full mile, will he finish in less than 5 minutes? How much less or how much more?

During a recent training session, Chris was able to run three miles in 21 minutes while Jerry was able to run the same distance in 23 minutes time. Which person was the faster runner? How many seconds faster per mile?

In the State Legislature, Representative Olson is pushing a new college loan program. 50 minority and 80 majority party representatives present today are in favor of the bill. One hundred representatives present today are unalterably opposed to the bill. Following the Governor's impassioned attack on the bill, enough uncommitted votes "switched" to opposed so that the same number of representatives are pledged against the bill as are pledged in favor of it. No other vote switches took place. If there are still 120 uncommitted representatives on this bill, how many were uncommitted before the Governor's speech?

Ms. Carpenter is trying to figure out how much money she needs to get her classroom of children to the Capitol Square for a tour. At the last minute school bus drivers have gone on strike. Mr. Cruz, the school principal, has authorized Mrs. Carpenter to take the 23 children in her class, herself, and 3 chaperones on the A metro line, which travels directly from their school to Capitol Square. All of the children are between the ages of 5 and 7, Ms. Carpenter and 2 of the chaperones are adults under the age of 64, while the remaining chaperone is a senior citizen. As the drivers' strike occurred at the last minute, all fares must be paid in cash. How much total cash will be needed to get everyone to the tour site and then back to the school on this day? (Refer to the bus schedule for bus fee information.)

Appendix C -- Skills of Cooperative Problem Solving: Examples and Counterexamples

I. Obtaining Information or Help from Others

Asking: Questioning others; Asking for help, information or suggestions

Attending: Watching, listening, and learning from others.

Clarifying: Rephrasing another's idea, to make sure you understand it

Using: Elaborating, extending, or acting upon another problem solver's idea

Consensus Testing: Inviting partner to agree or disagree with action or idea

Making Rules: Setting rules for working together, such who will do what.

II. Giving Information or Assistance

Replying: Answering others' questions

Monitoring: Initiating correction or registering disagreement

Supplying data: Giving information or opinion

Justifying: Explaining, giving reasons for action or opinion

Suggesting: Suggesting an action or interpretation

Supporting: Cheerleading, expressing positive feelings toward work or other.

Agreeing: Concurring with statement of fellow worker.

III. Non cooperative behaviors

Criticizing: Focusing frequently on what others are doing wrong

Being negative: Expressing bad attitude toward task or fellow workers

Isolating: Working alone in an uncooperative manner

Dominating: Insisting upon and imposing one's own method or opinion

Not caring: Dropping out and not paying attention

C1